TUGAS AKHIR – TE141599

PERANCANGAN SISTEM KENDALI GERAK LATERAL *WAY-TO-WAY POINT* UAV *QUADCOPTER* MENGGUNAKAN KONTROLER PID *FUZZY*

Rheco Ari Prayogo
NRP 2213 106 021

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.
Eka Iskandar, ST., MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – **TE141599**

DESIGN SYSTEM OF LATERAL MOTION CONTROL WAY-TO-WAY POINT UAV QUADCOPTER USING PID FUZZY CONTROLLER

Rheco Ari Prayogo
NRP 2213 106 021

Advisor
Ir. Josaphat Pramudijanto, M.Eng.
Eka Iskandar, ST., MT.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

PERANCANGAN SISTEM KENDALI GERAK LATERAL *WAY-TO-WAY POINT* UAV *QUADCOPTER* MENGGUNAKAN KONTROLER PID *FUZZY*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Josaphat Pramudijanto, M.Eng.	**Eka Iskandar, ST., MT.**
NIP.196210051990031003	NIP.198005282008121001

SURABAYA
JANAURI, 2016

DESIGN SYSTEM OF LATERAL MOTION CONTROL WAY-TO-WAY POINT UAV QUADCOPTER USING PID FUZZY CONTROLLER

Name **: Rheco Ari Prayogo**
***Supervisor* 1** **: Ir. Josaphat Pramudijanto, M.Eng.**
***Supervisor* 2** **: Eka Iskandar, ST., MT.**

ABSTRACT

Quadcopter is one type of unmanned aircraft where capable performing of rotational and translational motion. Lateral motion way-to-way point is translational motion where quadcopter will move forward, backward, and sideways to point to point of the field of coordinate axes x, y, and z. In this final project, trajectory plan is quadrilateral, so quadcopter will move with stable routed using Fuzzy PID controller.

Design Fuzzy PID control system is used to maintain the lateral motion quadcopter. In this case, auto-tuning fuzzy will take action that is used to determine the parameters kp, ki, and kd when quadcopter move down the track. Due to the plant parameters are always changing according to the circumstances, the Fuzzy-PID controller here is expected to maintain stable quadcopter real time while moving from the starting point to get back to the end point.

From the test, obtained from the tuning parameter values kp, ki, and kd had a good response even though there is an error in the translation of the X-axis and Y-axis of ± 0,01 cm and 0,035 cm ± as well as to achieve a steady-state is delayed by ± 2,2 seconds.

Keywords *: Quadcopter, Way-to-Way Point Motion, Lateral Motion, Auto Tuning,* PID *Fuzzy Controller.*

PERANCANGAN SISTEM KENDALI GERAK LATERAL *WAY-TO-WAY POINT* UAV *QUADCOPTER* MENGGUNAKAN KONTROLER PID *FUZZY*

Nama : **Rheco Ari Prayogo**
Dosen Pembimbing 1 : **Ir. Josaphat Pramudijanto, M.Eng.**
Dosen Pembimbing 2 : **Eka Iskandar, ST., MT.**

ABSTRAK

Quadcopter adalah salah satu jenis pesawat tanpa awak yang mampu melakukan gerak rotasi dan translasi. Gerak lateral *way-to-way point* merupakan gerak translasi dimana *quadcopter* akan bergerak maju, mundur, dan menyamping menuju titik – titik bidang sumbu koordinat x, y, dan z yang telah ditentukan. Dalam tugas akhir ini titik koordinat yang dibuat membentuk lintasan persegi, sehingga *quadcopter* akan bergerak menempuh lintasan tersebut dengan stabil menggunakan kontroler PID *Fuzzy*.

Perancangan sistem kontrol PID *Fuzzy* digunakan untuk mempertahankan gerak lateral *quadcopter*. Dalam hal ini *fuzzy* melakukan aksi *auto tuning* yang digunakan untuk menentukan parameter kp, ki, dan kd saat *quadcopter* bergerak menyusuri lintasan. Dikarenakan parameter *plant* yang selalu berubah-ubah sesuai dengan keadaan, maka kontroler PID-*Fuzzy* disini diharapkan mampu menjaga kestabilan *quadcopter* saat bergerak dari titik awal sampai kembali ke titik akhir secara *real time*.

Dari hasil pengujian didapatkan nilai parameter dari *tuning* kp, ki, dan kd memiliki respons yang baik meskipun terdapat kesalahan pada translasi sumbu X dan sumbu Y sebesar ± 0,01 cm dan ± 0,035 cm serta untuk mencapai *steady-state* masih terlambat sebesar ± 2,2 detik.

Kata Kunci: *Quadcopter,* Gerak *Way-to-Way Point*, Gerak *Lateral, Auto Tuning*, Kontrol PID *Fuzzy*.

KATA PENGANTAR

Segala puji kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya dalam usaha dan kerja keras sehingga penulis dapat menyelesaikan Tugas Akhir dengan judul :

"Perancangan Sistem Kendali Gerak Lateral *Way-toWay Point* UAV *Quadcopter* Menggunakan Kontroler PID *Fuzzy*"

Tugas Akhir ini disusun guna memenuhi persyaratan untuk menyelesaikan studi Teknik Elektro di Bidang Studi Teknik Sistem Pengaturan, Program Sarjana Teknik Elektro, Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya.

Dalam penyusunan laporan Tugas Akhir ini, penulis banyak mendapatkan bantuan dan dukungan dari berbagai pihak. Oleh karena itu, penulis dengan tulus ikhlas menyampaikan banyak terima kasih kepada:

1. Bapak Ir. Josaphat Pramudijanto, M.Eng. dan Eka Iskandar, ST., MT. sebagai dosen pembimbing penulis, atas segala kesabaran dan kesediaannya meluangkan waktu untuk membimbing serta memberi ilmu yang bermanfaat sehingga Tugas Akhir ini dapat terselesaikan.
2. Orang tua dan juga keluarga yang selalu mendukung sehingga Tugas Akhir ini dapat terselesaikan.
3. Dan semua pihak yang tidak dapat penulis sebutkan satu persatu yang telah memberi dorongan dan bantuan dalam menyelesakan Tugas Akhir ini baik secara langsung maupun tidak langsung.

Dengan segala kerendahan hati, kami berharap apa yang ada dalam buku Tugas Akhir ini dapat bermanfaat, dan berguna sebagai sumbangan pikiran bagi kita semua dalam berprestasi turut mengisi pembangunan Bangsa dan Negara.

Surabaya, Januari 2016

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

1.1 Latar Belakang

Unmanned Aerial Vehicle (UAV) atau pesawat terbang tanpa awak adalah salah satu robot penjelajah udara yang memiliki sistem kendali manual dan otomatis. Diantara robot terbang yang telah di kembangkan di Indonesia adalah *fixed wing, tricopter, quadcopter, hexacopter,* dan *octocopter*. Masing-masing robot terbang memiliki karakteristik yang berbeda sesuai dengan desain mekanis dan sistematisnya. Dalam penelitian tugas akhir ini jenis pesawat yang digunakan adalah *quadcopter*. Robot terbang ini memiliki empat baling-baling penggerak yang diposisikan tegak lurus terhadap bidang datar. Masing-masing rotor (baling-baling dan motor penggeraknya) menghasilkan gaya angkat dan memiliki jarak yang sama terhadap pusat massa pesawat. Dengan gaya angkat masing-masing rotor sebesar lebih dari seperempat berat keseluruhan, memungkinkan *quadcopter* untuk terbang. Kecepatan *quadcopter* tergantung pada kekuatan motor dan berat *quadcopter* itu sendiri. Untuk menghindari terjadinya momen putar pada *body*, arah putaran baling-baling dan kecepatan pada setiap rotornya berbeda. Terdapat 2 rotor yang bergerak searah jarum jam (*clockwise*) dan 2 rotor yang bergerak berlawanan arah jarum jam (*counter clockwise*). Kondisi ini memungkinkan *quadcopter* untuk bergerak translasi tegak lurus terhadap bidang. Namun keadaan rotor yang bergerak ini tergantung dengan kondisi yang di inginkan sesuai dengan kecepatan dan arahnya (*manuver*).

Gaya yang dihasilkan dari perputaran rotor disebut *rotorcraft*, menyebabkan *quadcopter* bergerak translasi vertikal akibat aliran udara yang dihasilkan perputaran propeler. Oleh karena tercipta gerak translasi vertikal tersebut, *rotorcraft* dapat melakukan pergerakan *landing* ataupun *take-off* secara vertikal sehingga tidak menyulitkan pengendali karena tidak dibutuhkan area yang luas sebagai landasan pacu.

Ada dua tipe konfigurasi *quadcopter*, yaitu konfigurasi tipe *plus* (+) dan konfigurasi tipe menyilang (X). Pada prinsipnya hampir sama, hanya bentuknya yang berbeda. Untuk mengatur *quadcopter* agar bergerak manuver atau mengatur laju / ketinggian, cukup memainkan kecepatan dan arah putaran masing-masing baling-baling. Pemberian

kecepatan yang sama dan arah yang berlawanan pada masing-masing baling-baling, maka pesawat akan naik dan mencapai ketinggian tertentu. Untuk bergerak turun adalah kebalikannya, yaitu menurunkan kecepatan putar masing-masing baling-baling secara simultan. Sedangkan dalam mengatur gerakan *manuver* maka yang dilakukan adalah mengatur kecepatan dan arah rotor sesuai tipe konfigurasi yang digunakan.

Ada tiga macam kategori yang harus diselesaikan dalam pengembangan robot terbang, yaitu efisiensi aerodinamika, peningkatan pembebanan, dan masalah kontrol dan stabilitas. *Quadcopter* dibagi dalam dua macam gerak terbang, yaitu gerakan longitudinal yang terbagi dalam tiga fase utama, yaitu *take-off* (lepas landas), *hovering* (melayang) dan *landing* (pendaratan), serta fase penerbangan gerakan lateral. Dari kedua gerakan terbang tersebut, gerakan lateral (terutama aplikasi *way-to-way point*) merupakan fase yang paling kritis dan masih perlu dilakukan banyak pengembangan.

Gerak translasi pada *quadcopter* di tentukan oleh resultan gaya dan sudut-sudut pada *quadcopter*. Jika dilakukan pengendalian gerak translasi, maka dari pernyataan sebelumnya dapat diambil hipotesa bahwa sinyal kontrol harus bergantung pada evaluasi gerak translasinya dan besar sudut-sudut *quadcopter*. Terdapat beberapa metode yang digunakan untuk menyelesaikan permasalahan tersebut. Salah satu Metode yang akan digunakan untuk mengatur dan menjaga kestabilan gerak lateral *quadcopter* dalam penelitian ini adalah kontroler PID-*Fuzzy.*

1.2 Permasalahan

Gerak *lateral* merupakan gerak *quadcopter* secara horizontal pada ketinggian atau gerak translasi, gerakan ini sangat vital untuk memenuhi kebutuhan *quadcopter* dalam mencapai *way-to-way point* yang telah ditentukan sehingga diperlukan kontroler untuk menstabilkannya. Perubahan nilai parameter *plant* menuntut kontroller mampu menjaga kestabilannya. Hal ini di lakukan karena parameter *plant* yang dihasilkan saat menyusuri lintasan berubah-ubah sesuai dengan karakteristik *plant*. Untuk mengatasi permasalahan ini, maka di gunakan metode *tuning fuzzy* agar nilai parameter kontroller PID berubah-ubah sesuai dengan parameter *plant* saat *quadcopter* bergerak melintasi lintasan. *Error* dan *delta error* sebagai masukan logika *fuzzy* nantinya akan berpengaruh terhadap nilai parameter kontroller PID. Hasil akhir

dari penelitian ini adalah mendesain kontroler agar dapat melakukan gerak lateral (*way to way point*) menuju titik-titik yang telah ditentukan dengan stabil.

1.3 Batasan Masalah

Secara realita, masalah yang ada dalam kontrol gerak *quadcopter* sangat banyak sekali. Untuk itu diperlukan batasan untuk mencari solusi yang tepat pada masalah yang ada. Batasan yang dilakukan antara lain :

1. Model UAV yang digunakan adalah jenis *quadrotor* dengan konfigurasi *propeller* berbentuk "*plus* (+)".
2. Titik *origin Euler-frame* dan *Body-frame* berhimpit.
3. Pemodelan dari *quadcopter* menggunakan pemodelan fisik yang akan disimulasikan menggunakan simulink Matlab..
4. Kestabilan yang dikontrol adalah sudut *pitch* dan *roll* pada gerak rotasi serta posisi sumbu X dan sumbu Y tanpa memperhitungkan perubahan ketinggian dari *quadcopter*.
5. Kontroler yang digunakan adalah kontroler PID untuk gerak rotasi dan kontroler PID-*Fuzzy* untuk gerak translasi.
6. Lintasan yang digunakan dalam gerak lateral berbentuk kotak dengan titik koordinat yang telah di tentukan.
7. Metode *Tuning Fuzzy* digunakan untuk menjaga kestabilan *quadcopter* saat melakukan gerak translasi. Hal ini dilakukan karena nilai parameter PID akan berubah-ubah sesuai dengan perubahan *plant*.

1.4 Tujuan

Tujuan dari penelitian Tugas Akhir ini adalah merancang kontroler PID-*Fuzzy* agar *quadcopter* mampu melakukan gerak lateral *way to way point* dengan menjaga kestabilan sudut *pitch* dan *roll* pada sumbu X dan sumbu Y sesuai *set point (*membuat *quadcopter* bergerak dengan lintasan yang diinginkan).

1.5 Sistematika Penulisan

Sistematika penulisan yang diterapkan pada buku tugas akhir ini terbagi menjadi lima bab, yaitu:

BAB I PENDAHULUAN

Pada bab pendahuluan membahas latar belakang, permasalahan, batasan masalah, tujuan, sistematika penulisan dan relevansi.

BAB II DASAR TEORI

Berisi teori-teori yang berkaitan dengan dinamika *plant*, aktuator, sensor dan kontroler yang digunakan dalam implementasi dan metode kontrol yang diterapkan.

BAB III PERANCANGAN SISTEM

Bagian ini berisi spesifikasi sistem, identifikasi kebutuhan, desain mekanik dan elektrik sistem, desain kontroler yang akan digunakan, diagram blok dari sistem secara keseluruhan, dan komponen-komponen yang akan digunakan dalam implementasi.

BAB IV PENGUJIAN DAN ANALISIS

Bab ini berisi hasil simulasi dengan menggunakan metode kontrol yang diterapkan dan analisa terhadap respon yang dihasilkan secara simulasi maupun implementasi.

BAB V PENUTUP

Kesimpulan dari seluruh pengerjaan tugas akhir dan saran untuk perbaikan dan pengembangan selanjutnya disajikan pada bab penutup.

1.6 Relevansi

Hal-hal yang dihasilkan dari tugas akhir ini diharapkan dapat menjadi referensi atau acuan untuk pengembangan *UAV quadcopter*, dan dijadikan pembanding untuk beberapa metode yang diterapkan nantinya.

BAB II
TEORI DASAR

Pada bab ini akan dibahas dasar teori yang diperlukan untuk analisis, simulasi dan rencana implementasi sistem. Secara garis besar, berisi teori analisis gerak dari *quadcopter*, kontroler PID-*Fuzzy*, dan komponen aktuator serta sensor yang di gunakan pada *plant quadcopter*.

2.1 Tinjauan Pustaka [1]

Quadcopter atau *quadrotor* adalah salah satu jenis pesawat tanpa awak yang memiliki empat motor dan dilengkapi dengan empat buah propeller pada masing-masing motornya yang digunakan untuk terbang dan bermanuver. Dengan empat buah motor ber-*propeller*, *quadcopter* dapat terbang lebih stabil dan tenang.

2.1.1 Sejarah

Quadcopter pertama kali dikembangkan oleh seorang berkewarganegaraan Perancis kelahiran tahun 1884 dari Ecole Centrale Paris yaitu Etienne Oemichen. Pada 18 Februari 1921 dia berhasil menerbangkan sebuah helikopter, lalu disusul pada tahun 1922 dia berhasil menerbangkan Oemichen No.2, yaitu salah satu prototipe *quadcopter* yang dapat terbang dengan menggunakan empat rotor dan delapan propeler (5 propeler untuk menstabilkan gerak dan 1 *propeller* digunakan untuk perpindahan arah muka pesawat sedangkan 2 *propeller* sisa digunakan untuk gerak maju).

Gambar 2.1 Prototipe Oemichen No.2

Tak jauh dari karya Oemichen, pada awal 1920an. Seorang bernama George de Bothezat juga menciptakan sebuah *quadcopter* yang digunakan untuk riset dari tentara nasional Amerika Serikat. *Quadcopter*

tersebut juga dijuluki sebagai Jerome-de Bothezat *Flying Octopus* dengan ukuran 20m x 20m x 3m.

Gambar 2.2 *Quadcopter* Bothezat

Perkembangan *quadcopter* sebagai robot terbang dimulai oleh Bouabdallah dan berhasil membuat *quadcopter* yang dinamakan OS4 seperti pada Gambar 2.3.

Penelitian *quadcopter* dengan model mekanik yang kecil semakin berkembang dengan mulai diciptakannya beberapa *quadcopter* seperti, X-4 FlyerMark (Australian National University), STARMAC (Stanford University), dan lain-lain.

Gambar 2.3 OS4 X-*flyer*

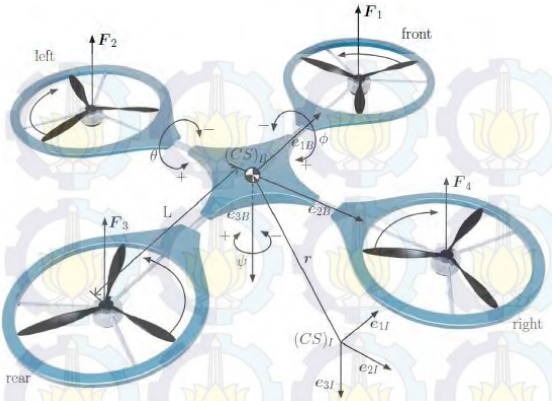
Gambar 2.4 *Quadcopter* STARMAC

Penelitian *quadrotor* yang berukuran kecil menjadi menarik dan semakin banyak variasi kontrol dan objektif kontrol dikembangkan.

2.1.2 Konsep Dasar

Quadrotor yang digunakan sebagai robot terbang kecil memiliki model mekanik yang terdiri dari empat buah motor yang dipasang pada sumbu silang simetris. Keempat buah motor berputar dengan kecepatan yang sama ditiap *propeller* dan digunakan sebagai penggerak baling-baling untuk menghasilkan gaya angkat. Setiap *propeller* pada *quadcopter* ini diputar oleh satu motor elektrik, sehingga terdapat empat motor sebagai aktuator untuk menghasilkan gaya angkat dari *quadcopter*.

Dengan batasan menggunakan karakteristik motor dan *propeller* yang relatif sama, maka kondisi melayang yang stabil akan diperoleh kecepatan motor yang sama ditiap *propeller*. Pergerakan *quadrotor* bergantung pada putaran masing-masing *propeller*. Untuk keseimbangan terbang *quadcopter*, maka konfigurasi putaran antara motor depan dan belakang harus sama dan berlawan arah dengan putaran motor kiri dan kanan. Pada penelitian Tugas Akhir yang dilakukan, ditentukan bahwa putaran motor depan dan belakang searah jarum jam, sedangkan putaran rotor kanan dan kiri berputar berlawanan arah jarum jam.Pada Gambar 2.5, struktur *quadcopter* ditunjukkan bahwa terdapat empat masukan yang terdapat pada *quadcopter*, yaitu *roll*, *pitch*, *yaw* dan *thrust*.

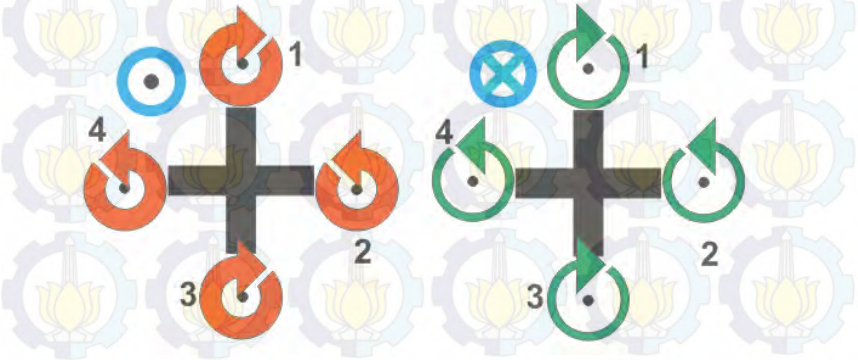


Gambar 2.5 Struktur *Quadcopter*

Pengaturan kecepatan putaran masing-masing motor akan menghasilkan empat gerakan dasar yang memungkinkan *quadcopter* untuk mencapai ketinggian dan sikap *(altitude)* tertentu, yaitu :

1. Gaya *Thrust*

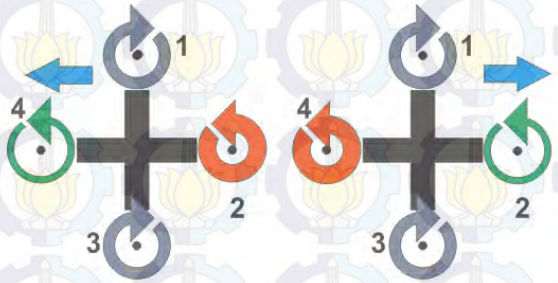
Pada gerakan ini, putaran keempat motor berputar dengan nilai yang sama. Jika keempat rotor berputar semakin cepat atau lambat dengan kecepatan yang sama akan menghasilkan percepatan vertikal. Dapat dilihat dari Gambar 2.6, *propeller* depan (1) dan belakang (3) berputar searah jarum jam, sedangkan *propeller* kanan (2) dan kiri (4) berputar melawan arah jarum jam. Gerakan *thrust* ditampilkan pada Gambar 2.6.



Gambar 2.6 Gerakan *Thrust*

2. Gerakan R*oll*

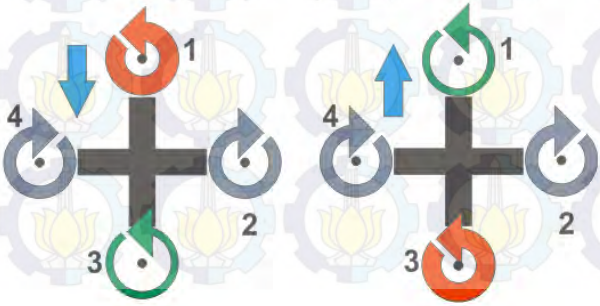
Pergerakan ini dilakukan dengan cara mempercepat atau memperlambat putaran dari motor sebelah kiri atau kanan. Pergerakan ini akan menghasilkan manuver rotasi ke kanan atau ke kiri tergantung dari penurunan dan peningkatan kecepatan putaran motor yang tersaji pada Gambar 2.7. Gerakan ini bergerak dengan acuan pada sumbu X.



Gambar 2.7 Gerakan R*oll*

3. Gerakan P*itch*

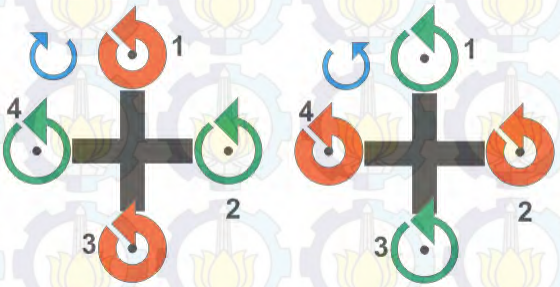
Pola dari pergerakan ini memiliki prinsip yang sama dengan gerakan *roll*. Agar menghasilkan gerakan *pitch*, maka kecepatan putar motor depan atau belakang saling berlawanan. Pergerakan ini dilakukan dengan cara menambahkan (atau mengurangi) kecepatan putar motor 1 pada *quadcopter* dan bersamaan dengan itu menurunkan (atau menaikkan) kecepatan motor 3. Gerakan ini bergerak dengan acuan pada sumbu Y. Gerakan *pitch* ditunjukkan pada Gambar 2.8.



Gambar 2.8 Gerakan *Pitch*

4. Gerakan Y*aw*

Pergerakan ini dilakukan untuk memutar posisi *quadrotor* dengan inti tetap berada pada posisi atau titik yang sama. Gerakan *yaw* dicapai dengan mempercepat atau memperlambat kecepatan putaran motor yang berlawanan arah jarum jam, serta memperlambat atau mempercepat kecepatan putaran motor yang berputar searah jarum jam. Pola pergerakan *yaw* akan diperoleh dengan percepatan sudut yang ditampilkan pada Gambar 2.9.



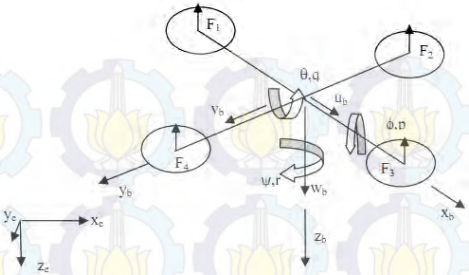
Gambar 2.9 Gerakan *Yaw*

2.1.3 Model *Quadcopter*

Pemodelan secara fisik terbilang kompleks dan apabila tanpa adanya asumsi yang digunakan untuk menyederhanakan persamaan, perhitungannya menjadi tidak praktis. Beberapa asumsi yang digunakan dalam pemodelan ini adalah :

1. Struktur dari *quadcopter* dikatakan *rigid*
2. Struktur dari *quadcopter* dikatakan simetris
3. Struktur dari *propeller* dikatakan *rigid*
4. Gaya *thrust* dan *drag* adalah proporsional dengan kuadrat dari kecepatan *propeller*
5. Keadaan model adalah keadaan ketika *hovering*

Quadcopter memiliki 6 *defree of freedom (DOF)*. Untuk mendeskripsikan gerakan dari 6 *DOF rigid-body* digunakan dua buah *frame* referensi yaitu *earth inersial reference* (*E-frame*) dan *body fixed reference* (*B-frame*)



Gambar 2.10 Ilustrasi *B-frame* Terhadap *E-frame*

2.2 Kontroler PID (*Proportional, Integral, Derivative*) [2]

Setiap kekurangan dan kelebihan dari masing-masing kontroler Proporsional, Integral, dan Diferensial dapat saling menutupi dengan menggabungkan ketiganya menjadi kontroler proporsional *plus* integral *plus* diferensial (kontroler PID). Elemen-elemen kontroler P, I, dan D masing-masing secara keseluruhan bertujuan untuk mempercepat reaksi sebuah sistem, menghilangkan *offset* dan menghasilkan perubahan awal besar.

2.2.1 Kontroler PID Standar

Hubungan sinyal kesalahan dan sinyal kontrol pada kontroler tipe-PID standar dapat dinyatakan dengan Persamaan (2.1).

$$u(t) = K_p\left[e(t) + \frac{1}{\tau_I}\int e(t)dt + \tau_D \frac{d}{dt}e(t)\right] \tag{2.1}$$

Atau dalam bentuk fungsi alih,

$$\frac{U(s)}{E(s)} = K_p(1 + \frac{1}{\tau_I s} + \tau_D s) \tag{2.2}$$

Atau

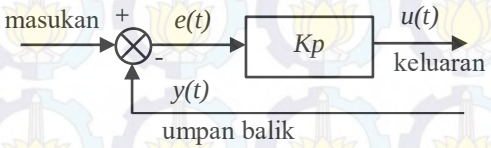
$$\frac{U(s)}{E(s)} = \frac{K_p(\tau_I \tau_D s^2 + \tau_I s + 1)}{\tau_I s} \tag{2.3}$$

2.2.2 Kontroler *Proportional*

Kontroler proposional memiliki keluaran yang sebanding atau proposional dengan besarnya sinyal kesalahan (selisih antara besaran yang diinginkan dengan harga aktualnya). Secara lebih sederhana dapat

dikatakan, bahwa keluaran kontroler proporsional merupakan perkalian antara konstanta proporsional dengan masukannya.

Perubahan pada sinyal masukan akan segera menyebabkan sistem secara langsung mengubah keluarannya sebesar konstanta pengalinya. Blok diagram kontroler proporsional ini ditunjukkan pada Gambar 2.11. Persamaan matematis kontroler proporsional ditunjukkan pada Persamaan 2.4.



Gambar 2.11 Diagram Blok Kontroler Proporsional

$$u(t) = K_p e(t) \tag{2.4}$$

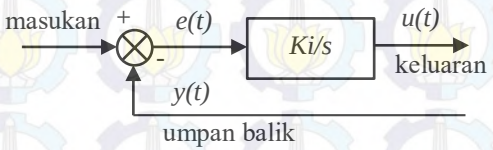
Dimana K_P adalah konstanta proporsional. Ciri-ciri kontroler proporsional harus diperhatikan ketika kontroler tersebut diterapkan pada suatu sistem. Secara eksperimen, pengguna kontroler proporsional harus memperhatikanketentuan-ketentuan berikut ini:

1. Jika nilai Kp kecil, kontroler proporsional hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan respon sistem yang lambat.
2. Jika nilai Kp dinaikkan, respon sistem menunjukkan semakin cepat mencapai keadaan mantapnya *(steady state)*.
3. Jika jika nilai Kp diperbesar sehingga mencapai harga yang berlebihan, akan mengakibatkan sistem bekerja tidak stabil, atau respon sistem akan berosilasi.

2.2.3 Kontroler *Integral*

Kontroler integral berfungsi menghasilkan respon sistem yang memiliki kesalahan keadaan tunak *(error steady state)* nol. Jika sebuah *plant* tidak memiliki unsur *integrator*, kontroler proporsional tidak akan mampu menjamin keluaran sistem dengan kesalahan keadaan tunaknya nol. Dengan kontroler integral, respon sistem dapat diperbaiki, yaitu mempunyai kesalahan keadaan mantapnya nol. Gambar 2.12,

menampilkan diagram blok kontroler integral dengan masukan E(s) dan keluaran U(s).



Gambar 2.12 Diagram Blok Kontroler Integral

Kontroler integral memiliki karakteristik seperti halnya sebuah integral. Keluaran kontroler sangat dipengaruhi oleh perubahan yang sebanding dengan nilai sinyal kesalahan. Jika sinyal kesalahan tidak mengalami perubahan, keluaran akan menjaga keadaan seperti sebelum terjadinya perubahan masukan. Persamaan matematis kendali integral ditunjukkan pada Persamaan 2.5.

$$u(t) = \frac{1}{T_i}\int_0^t e(t)dt \tag{2.51}$$

Konsekuensi dengan memperbesar *gain* integral atau memperkecil konstanta waktu integral dapat membuat sistem lebih cepat "mengejar" nilai keadaan tunak, namun sistem cenderung akan berosilasi. Dalam keadaan seperti ini, diperlukan sebuah metode yang dapat memperkecil amplitudo osilasi dan nilai *overshoot* dari respon.

Ketika digunakan, kontroler integral mempunyai beberapa karakteristik, antara lain :

1. Keluaran kontroler membutuhkan selang waktu tertentu, sehingga kontroler integral cenderung memperlambat respon.
2. Ketika sinyal kesalahan berharga nol, keluaran kontroler akan bertahan pada nilai sebelumnya.
3. Jika sinyal kesalahan tidak berharga nol, keluaran akan menunjukkan kenaikan atau penurunan yang dipengaruhi oleh besarnya sinyal kesalahan dan nilai *Ki*.
4. Konstanta integral *Ki* yang berharga besar akan mempercepat hilangnya *offset*. Tetapi semakin besar nilai konstanta *Ki* akan mengakibatkan peningkatan osilasi dari sinyal keluaran kontroler.

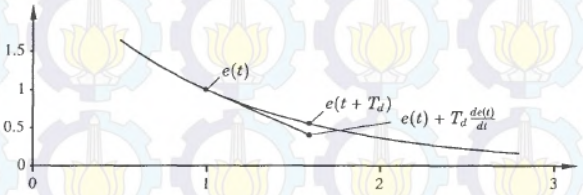
2.2.4 Kontroler *Derivative*

Secara intuitif, respon sistem yang berosilasi disebabkan pada beberapa hal. Proses dinamik dari sebuah *plant* menyebabkan respon sebuah *plant* tidak langsung berubah dengan adanya perubahan sinyal kontrol, namun membutuhkan waktu proses. Waktu ini akan membuat sistem kontrol mengalami keterlambatan untuk mengkoreksi kesalahan. Untuk itu dibutuhkan kontroler yang dapat memprediksi kesalahan dari sebuah sistem seperti pada Gambar 2.15. Nilai kesalahan prediksi dapat diturunkan dari persamaan taylor sebagai berikut:

$$e(t+Td) \approx e(t)+T_d \frac{de(t)}{dt} \tag{2.6}$$

Dengan mengatur gradien dari garis prediksi, maka akan diperoleh kesalahan prediksi yang lebih akurat, dan persamaan menjadi:

$$u(t) = K_p(e(t)+T_d \frac{de(t)}{dt}) \tag{2.7}$$

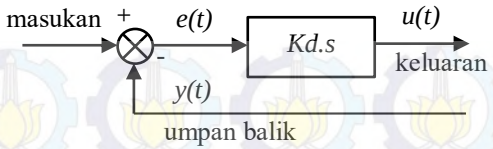


Gambar 2.13 Prediksi Kesalahan

Sebagai validasi, seperti yang telah dipaparkan bahwa osilasi dari respon sistem terjadi karena sinyal kontrol berlebihan yang diberikan ke *plant*. Maka diperlukan mekanisme untuk mengurangi nilai sinyal kontrol yang berlebihan tersebut. Kontroler derivatif yang ditambahkan pada kontroler proporsional dapat mengurangi aksi kontrol yang berlebihan.

Dari Persamaan 2.33, dapat dilihat bahwa keadaan awal saat sistem memiliki kesalahan positif maka kontroler proporsional akan berusaha mengejar nilai *setpoint*. Dengan adanya respon tersebut, maka perubahan kesalahan proporsional dengan negatif kesalahan. Hal ini akan membuat kontroler derivatif akan berusaha meredam aksi kontroler proporsional yang berlebihan.

Berkurangnya nilai sinyal kontrol akan berdampak pada kecepatan respon sistem yang semakin berkurang. Gambar 2.14 menampilkan diagram blok kontrol derivative.



Gambar 2.14 Diagram Blok Kontroler Derivatif

Ketika digunakan, kontroler derivatif mempunyai beberapa karakteristik, antara lain :

1. Kontroler ini tidak dapat menghasilkan keluaran bila tidak ada perubahan pada masukannya (berupa sinyal kesalahan).
2. Jika sinyal kesalahan berubah terhadap waktu, maka keluaran yang dihasilkan kontroler tergantung pada nilai K_d dan laju perubahan sinyal kesalahan.
3. Kontroler derivatif mempunyai suatu karakter untuk mendahului, sehingga kontroler ini dapat menghasilkan koreksi yang signifikan sebelum pembangkit kesalahan menjadi sangat besar. Jadi kontroler derivatif dapat mengantisipasi pembangkit kesalahan, memberikan aksi yang bersifat korektif, dan cenderung meningkatkan stabilitas sistem.

Berdasarkan karakteristik kontroler tersebut, kontroler derivatif umumnya dipakai untuk mempercepat respon awal suatu sistem, tetapi tidak memperkecil kesalahan pada keadaan tunaknya. Kerja kontroler derivatif hanyalah efektif pada lingkup yang sempit, yaitu pada periode peralihan. Oleh sebab itu kontroler derivatif tidak pernah digunakan tanpa ada kontroler lain sebuah sistem.

2.2.5 *Tuning* Eksperimen

Tuning eksperimen adalah proses yang dilakukan untuk mendapatkan hasil kontroler yang optimal dengan cara suatu percobaan. Inti dari *tuning* eksperimen adalah menentukan nilai dari tiga buah parameter yang terdapat pada kontroler PID yaitu konstanta proporsional (K_p), konstanta integral (K_i) dan konstanta diferensial (K_d). Ada beberapa metode yang dapat dilakukan dalam penalaan parameter kontroler PID dengan eksperimen seperti metode Ziegler-Nichols, metode Cohen-Coon dan metode empirik. Langkah atau acuan

penentuan parameter K_p, K_i, dan K_d diadopsi dari (Williams, 2006). Langkah metode tersebut adalah sebagai berikut:

1. Langkah awal gunakan kontroler proporsional terlebih dahulu, abaikan konstanta integratif dan derivatifnya dengan memberikan nilai nol pada integratif dan derivatif.
2. Tambahkan terus konstanta proporsional maksimum hingga keadaan stabil namun robot masih berosilasi.
3. Untuk meredam osilasi, tambahkan konstanta diferensial dengan membagi dua nilai proporsional, amati keadaan sistem robot hingga stabil dan lebih responif.
4. Jika sistem robot telah stabil, kontrol integral dapat menjadi *optional*, dalam artisan jika ingin mencoba-coba tambahkan kontrol integral tersebut, namun pemberian nilai integral yang tidak tepat dapat membuat sistem robot menjadi tidak stabil.
5. Nilai *sampling time* (waktu cuplik) juga mempengaruhi perhitungan PID, tentunya saat penggunaan kontrol integral dan diferensial.
6. Periksa kembali performa sistem hingga mendapatkan hasil yang memuaskan.

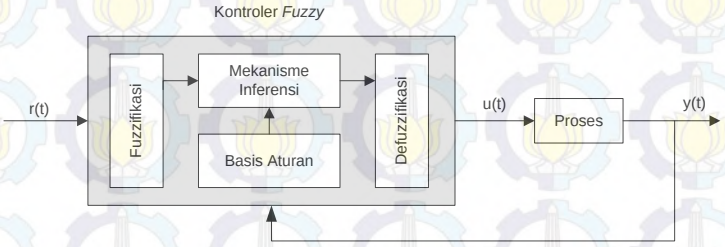
2.3 Metode Logika Fuzzy [3]

Sebelum konsep logika *fuzzy* diperkenalkan, orang telah mengenal konsep yang disebut logika klasik yang membagi sifat parameter menjadi dua hal yang berlawanan secara tegas, seperti benar atau salah, 0 atau 1. Konsep ini ternyata mempunyai kekurangan dalam penerapannya di kehidupan nyata, karena manusia lebih mengenal konsep linguistik yang menyatakan sesuatu dengan tidak eksak atau samar.

Konsep logika *fuzzy* mengubah konsep logika klasik menjadi konsep yang memetakan suatu variabel pada kemungkinan yang tidak eksak sehingga didapatkan sistem linguistik dan permasalahan yang tidak pasti atau tidak presisi serta permasalahan probabilitas.

2.3.1 Struktur Dasar Logika *Fuzzy*

Pada dasarnya struktur logika *fuzzy* tampak seperti Gambar 2.17 berikut :



Gambar 2.15 Struktur Logika *Fuzzy*

Gambar 2.15 menunjukkan struktur logika *fuzzy*. Fungsi dari bagian-bagian di atas adalah sebagai berikut:

1. Fuzzifikasi berfungsi untuk mentransformasikan sinyal *input* yang bersifat *crisp* (bukan *fuzzy*) ke himpunan *fuzzy* dengan menggunakan operator fuzzifikasi
2. Basis pengetahuan berisi basis data dan aturan dasar yang mendefinisikan himpunan *fuzzy* atas daerah-daerah *input* dan *output* dan menyusunnya dalam perangkat aturan kontrol.
3. Logika pengambilan keputusan merupakan inti dari logika *fuzzy* yang mempunyai kemampuan seperti manusia dalam mengambil keputusan. Aksi atur *fuzzy* disimpulkan dengan menggunakan implikasi *fuzzy* dan mekanisme inferensi *fuzzy*.
4. Defuzzifikasi berfungsi untuk mentransformasikan kesimpulan tentang aksi atur yang bersifat *fuzzy* menjadi sinyal sebenarnya yang bersifat *crisp* dengan menggunakan operator fuzzifikasi.

2.3.2 Fuzzifikasi

Fuzzifikasi adalah proses pemetaan *input* dan *output* sistem agar sesuai dengan himpunan *fuzzy*. Pemetaan digunakan dengan cara yang disebut fungsi keanggotaan (*membership function*). Ada banyak metode yang digunakan untuk proses fuzzifikasi, tetapi bentuk *triangular* dan *trapezoidal* yang paling banyak digunakan dalam pembentukan fungsi keanggotaan pada logika *fuzzy*. Hal ini dikarenakan metode bentuk

triangular dan *trapezoidal* lebih mudah diimplementasikan pada kontroler. Persamaan fuzzifikasi untuk metode *triangular* adalah sebagai berikut,

$$f(x;a,b,c)=\begin{cases}0 & ,x\le a\\ \dfrac{x-a}{b-a} & ,a\le x\le b\\ \dfrac{c-x}{c-b} & ,b\le x\le c\\ 0 & ,c\le x\end{cases} \tag{2.8}$$

Sedangkan untuk fuzzifikasi metode *trapezoidal* adalah sebagai berikut,

$$f(x;a,b,c)=\begin{cases}0 & ,x\le a\\ \dfrac{x-a}{b-a} & ,a\le x\le b\\ 1 & b\le x\le c\\ \dfrac{c-x}{c-b} & c\le x\le d\\ 0 & d\le x\end{cases} \tag{2.9}$$

2.3.3 Aturan Dasar *Fuzzy*

Aturan dasar *fuzzy* adalah kaidah dasar yang berisi aturan-aturan secara linguistik yang menunjukkan kepakaran terhadap *plant*. Banyak cara menunjukkan suatu kepakaran ke dalam aturan, format yang paling umum adalah sebagai berikut :

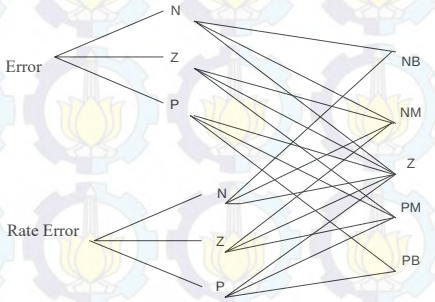
1. Format Aturan *IF-THEN*

   "IF premis THEN conclusion". Premis berupa fakta, dengan demikian dari kepakaran dapat diambil kesimpulan. Apabila pernyataannya lebih dari satu maka dapat digunakan logika "*AND*" atau "*OR*".

   Contoh penggunaan aturan *IF-THEN* sebagai berikut :
   IF error is N THEN output is NB (*Negative Big*)
   IF error is Z THEN output is Zero
   IF error is P THEN output is PB (*Positive Big*)

2. Format Hubungan

Pada dasarnya sama dengan aturan *IF-THEN* hanya saja tampilannya lebih sederhana karena menggunakan hubungan garis. Contoh dari penggunaan format hubungan dapat dilihat pada Gambar 2.18.



Gambar 2.16 Aturan Dasar *Fuzzy*

Gambar 2.16 menjelaskan NB adalah *Negative Big*, NM adalah *Negative Medium*, Z adalah *Zero*, PM adalah *Positive Medium*, dan PB adalah *Positive Big* seperti pada Tabel 2.1

3. Format *Tabular*

Tabel 2.1 menunjukkan Format Turbular lebih sederhana daripada format hubungan, variabel linguistik berada pada sisi luar dari tabel, sedangkan sisi dalam berisi dari keputusannya.

Tabel 2.1 Format Turbular

Error/Rate Error	*Negative*	*Zero*	*Positive*
Negative	NB	NM	Zero
Zero	NM	Zero	PM
Positive	Zero	PM	PB

2.3.4 Logika Pengambilan Keputusan

Inferensi *fuzzy* adalah suatu proses formulasi pemetaan *input* terhadap *output* dengan menggunakan logika *fuzzy*. Proses dari inferensi *fuzzy* melibatkan fungsi keanggotaan operator logika *fuzzy* dan aturan *if-then*. Terdapat dua metode inferensi yang paling dikenal, yaitu metode inferensi *Mamdani* dan metode *Takagi-Sugeno*. Metode inferensi *Mamdani* menggunakan fungsi keanggotaan *fuzzy* pada bagian *output*nya. Sehingga setelah proses aturan telah diterapkan, terdapat

himpunan *fuzzy* yang harus didefuzzifikasi. Umumnya proses defuzzifikasi berlangsung lebih lambat akibat proses komputasi pada *output*nya.

Metode *Takagi-Sugeno* menggunakan fungsi keanggotaan *output* yang linier atau berupa konstanta. Sedangkan dua bagian pada proses inferensi yaitu fuzzifikasi dan penerapan operator *fuzzy* sama dengan metode inferensi *Mamdani*. Bentuk aturan *fuzzy Takagi-Sugeno* memiliki bentuk sebagai berikut :

If input1 = x and input2 = y, then output is z = ax + by + c (2.10)

2.3.5 Defuzzifikasi

Defuzzifikasi adalah proses yang digunakan untuk mengubah kembali variabel *fuzzy* menjadi variabel nyata, atau dengan kata lain aksi pengaturan *fuzzy* yang masih berupa himpunan, diubah menjadi nilai nyata yang berupa nilai tunggal. Banyak metode yang dapat digunakan untuk proses defuzzifikasi, salah satu contohnya adalah metode *center of gravity*. Persamaan (2.53) adalah contoh defuzzifikasi dengan *center of gravity*.

$$\mu = \frac{\sum_{i-1}^{n} \{(masukan_i) \times keluaran\}}{\sum_{i=1}^{n} \{f(masukan_i)\}} \quad (2.11)$$

2.4 Fuzzy Mamdani [3]

Metode Mamdani sering juga dikenal dengan nama metode Max-Min**.** Metode ini diperkenalkan oleh Ebrahim Mamdani pada tahun 1975. Untuk medapatkan *output* diperlukan beberapa tahapan, antara lain:

2.4.1. Pembentukan Himpunan *Fuzzy*.

Pada Metode Mamdani, baik variabel *input* maupun variabel *output* dibagi menjadi satu atau lebih himpunan *fuzzy*.

2.4.2 Aplikasi Fungsi Implikasi

Pada Metode Mamdani, fungsi implikasi yang digunakan adalah fungsi minimal. Secara umum dapat dituliskan pada Persamaan 2.12.

$\mu A \cap B = \min (\mu A [x], \mu B [x,])$ (2.12)

2.4.3 Komposisi Aturan

Tidak seperti penalaran monoton, apabila sistem terdiri dari beberapa aturan, maka inferensi diperoleh dari kumpulan dan kolerasi antar aturan. Ada 3 metode yang digunakan dalam melakukan inferensi sistem *fuzzy*, yaitu *max*, *additive* dan *probabilistik* OR (probor).

1. Metode *Max (Maximum)*

   Metode *Max (Maximum)* mengambil solusi himpunan *fuzzy* diperoleh dengan cara mengambil nilai maksimum aturan, kemudian menggunakannya untuk memodifikasi daerah *fuzzy*, dan mengapilasikannya ke *output* dengan menggunakan operator OR *(union)*. Jika semua proposisi telah dievaluasi, maka *output* akan berisi suatu himpunan *fuzzy* yang merefleksikan kontribusi dari tiap-tiap proporsi. Secara umum dapat dituliskan pada Persamaan 2.13.

   $$\mu sf[Xi] \leftarrow (\mu sf[Xi], \mu kf[Xi]) \qquad (2.13)$$

   dengan :
   $\mu sf[xi]$ = nilai keanggotaan solusi *fuzzy* sampai *rule* ke - i
   $\mu kf[xi]$ = nilai keanggotaan konsekuen *fuzzy* sampai *rule* ke - i

2. Metode *Additive (Sum)*

   Metode *Additive (Sum)* mengambil solusi himpunan *fuzzy* diperoleh dengan cara melakukan *bounded-sum* terhadap semua *output* daerah *fuzzy*. Secara umum dituliskan ada persamaan 2.14.

   $$\mu sf[Xi] \leftarrow \min(\mu sf[Xi], \mu kf[Xi] + \mu kf[Xi]) \qquad (2.14)$$

   dengan:
   $\mu sf[xi]$ = nilai keanggotaan solusi *fuzzy* sampai *rule* ke – i
   $\mu kf[xi]$ = nilai keanggotaan konsekuen *fuzzy* sampai *rule* ke - i

3. Metode *Probabilistik OR (probor)*

   Metode *Probabilitik OR (probor)* mengambil solusi himpunan *fuzzy* diperoleh dengan cara melakukan *product* terhadap semua *output* daerah *fuzzy*. Secara umum dituliskan dengan :

$$\mu sf[Xi] \leftarrow -(\mu sf[Xi], \mu kf[Xi]) - (\mu sf[Xi] * \mu kf[Xi]) \quad (2.15)$$

dengan :
μsf [xi] = nilai keanggotaan solusi *fuzzy* sampai *rule* ke – i
μkf [xi] = nilai keanggotaan konsekuen *fuzzy* sampai *rule* ke - i

2.4.4 Penegasan (Defuzzifikasi)

Input dari proses *defuzzyfikasi* adalah suatu himpunan *fuzzy* yang diperoleh dari komposisi aturan-aturan *fuzzy*, sedangkan *output* yang dihasilkan merupakan suatu bilangan pada domain himpunan *fuzzy* tersebut.

BAB III
PERANCANGAN SISTEM

Pada bab ini akan dibahas mengenai perancangan sistem *quadcopter* meliputi spesifikasi, identifikasi kebutuhan, komponen mekanik dan elektronik, model matematik, identifikasi konstanta, perancangan kontroler PID-*Fuzzy*, dan simulasi sistem.

3.1 Spesifikasi Sistem

Pada bagian ini, akan dijelaskan mengenai spesifikasi *quadcopter* pada sisi hardware dan simulasi menggunakan software Matlab yang digunakan untuk menganalisa dan mengevaluasi perancangan kontroler yang akan di rancang. *Quadcopter* memiliki 6 derajat kebebasan (*degree of freedom*), terdiri dari 3 dof rotasi dan 3 dof translasi. Spesifikasi sistem yang diharapkan tercapai pada perancangan ini adalah :

1. *Quadcopter* mampu melakukan gerak lateral *way-to-way point* menyusuri sumbu X dan sumbu Y sesuai dengan titik koordinat yang telah di tentukan.
2. *Quadcopter* mampu mnejaga kestabilan sudut *roll* dan *pitch* selama melakukan gerak lateral.
3. *Quadcopter* dapat digerakkan secara manual dengan menggunakan *remote control*.
4. Parameter kontroler dapat berubah sesuai dengan perubahan parameter *plant.*
5. Kontroler dapat berfungsi sebagaimana mestinya.

3.2 Identifikasi Kebutuhan

Perancangan *quadcopter* harus memenuhi spesifikasi yang ada. Untuk memenuhi spesifikasi tersebut maka kebutuhan yang harus dipenuhi antara lain :

a. Desain *quadcopter* dibuat ringan dan simetris
b. Baterai yang digunakan minimal mampu membuat *quadcopter* terbang sekitar 10 menit.
c. Rangkaian mikrokontroler memiliki memori dan *port I/O* yang cukup untuk sensor, aktuator dan komunikasi.
d. Terdapat minimal 5 kanal yang terdiri dari 4 kanal untuk *input* sistem dan 1 kanal untuk pengaturan mode.
e. Data-data selama terbang dapat dikirimkan ke *ground station*.

3.3 Desain Mekanik *Quadcopter*

Desain mekanik yang baik akan mendukung pergerakan *quadcopter* menjadi lebih baik. Oleh karena itu *frame quadcopter* harus dibuat simetris dan cukup ringan agar mudah untuk diterbangkan. *Quadcopter* yang simetris dapat dilihat dari nilai momen inersia pada sumbu translasi maju dan menyamping memiliki nilai yang sama. Beberapa desain mekanik *quadcopter* disajikan pada Gambar 3.1 sebagai acuan dalam perancangan *quadcopter*.

Gambar 3.1 Contoh Desain Mekanik *Quadcopter*

3.3.1 *Frame* Taloon V1 [4]

Frame yang digunakan pada tugas akhir ini adalah *Frame* Taloon V1dari Turnigy. *Frame* ini berbahan dasar *carbon fiber* sehingga menjadikan *body quadcopter* kuat dan ringan sesuai dengan spesifikasi *quadcopter* yang digunakan. *Body quadcopter* berbentuk *plus* atau menyilang dengan penambahan motor dan *propeller* pada masing-masing ujungnya. Gambar 3.2 adalah tampilan dari *frame quadcopter* yang digunakan. Spesifikasinya adalah :

Berat	: 240 gram
Lebar	: 498 mm
Motor Bolt Holes	: 22 ~ 32 mm
Pasangan	: 4 ESC (*Electronic Speed Controller*), 4 Motor *Brushless*, dan 4 *Propeller*

Gambar 3.2 *Frame* Taloon V1 *Quadcopter*

3.3.2 *Propeller* HQPROP 10x4,5R [5]

Propeller adalah salah satu bagian mesin yang berfungsi sebagai alat penggerak mekanik. Dalam tugas akhir ini digunakan empat buah *propeller* dan berfungsi sebagai baling-baling pesawat yang menghasilkan gaya dorong ke atas (gaya angkat/*thrust*) sehingga menyebabkan pesawat dapat melakukan *take off*, *landing*, maupun *manuver*. Gambar 3.3 adalah tampilan *propeller* yang digunakan dalam tugas akhir ini dengan spesifikasi sebagai berikut :

Diameter : 45 cm
Jari-Jari : 22,5 cm

Gambar 3.3 *Propeller* HQPROP 10x4,5R

3.3.3 Alumunium

Penggunaan Aluminium berfungsi sebagai pelindung *propeller* dan *quadcopter* agar tetap aman saat melakukan aksinya. Aluminium didesain dan dirancang sedemikian rupa agar simetris dengan *frame body* dan tidak menjadi penghalang *quadcopter* untuk terbang. Alumunium yang digunakan ditampilkan pada Gambar 3.4.

Gambar 3.4 Aluminium Siku

Secara keseluruhan desain mekanik yang dibangun pada penelitian tugas akhir ini ditunjukkan pada Gambar 3.5.

Gambar 3.5 Desain Mekanik

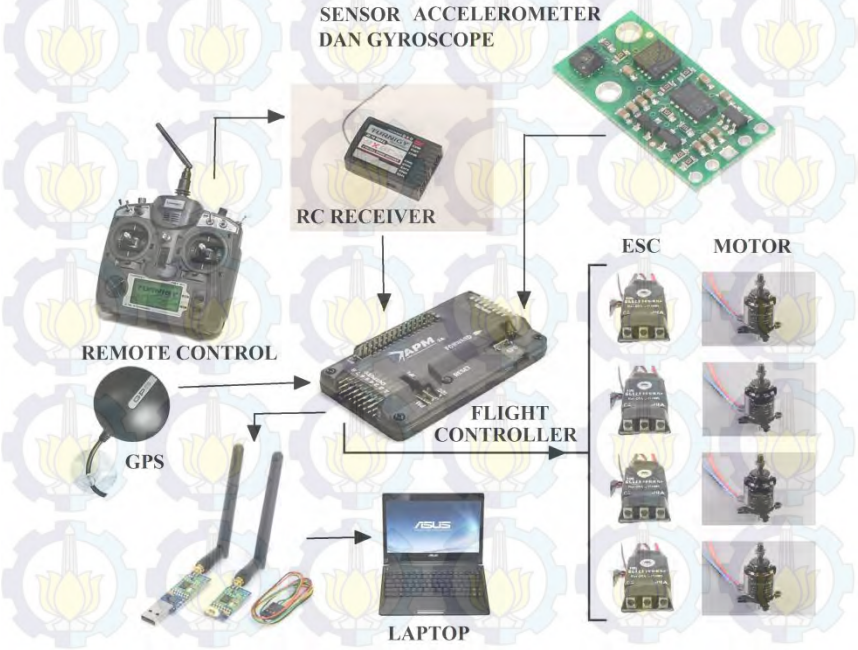
3.4 Desain Elektronik *Quadcopter*

Sistem elektronika tidak pernah lepas dari perancangan sistem, karena pada tahap inilah semua komponen sistem akan diintegrasikan agar dicapai sebuah keluaran yang diinginkan. *Quadcopter* terdiri atas sistem kontroler berupa APM Planner, sensor, sistem komunikasi, motor *brushless*, dan *remote control* yang digunakan sebagai acuan dalam menentukan gerak terbangnya. Sensor yang digunakan adalah sensor

sudut yang diukur menggunakan sensor *accelerometer*, dan terakhir sensor kecepatan sudut yang diukur menggunakan sensor *gyroscope*.

Rangkaian elektronika yang dirancang harus mampu menampung jumlah *input/output* sensor dan komponen-komponen yang digunakan sebagai penunjang terbang *quadcopter*. Beberapa komponen seperti sensor-sensor, *ESC (Electronic Speed Controller)*, dan motor *brushless* merupakan modul yang dapat langsung digunakan. Perancangan keseluruhan dari sistem elektronika dari *quadcopter* ditunjukkan pada Gambar 3.6.

Board kontroler yang dipakai adalah APM Planner 2.6 yang sudah dilengkapi beberapa sensor dan *port* yang memudahkan untuk penggunaan sebagai komunikasi data. *Board* APM Planner 2.6 dilengkapi dengan ATMega 2560 sebagai mikrokontroler dari *flight controller* tersebut.



Gambar 3.6 Rancangan Sistem Elektronika *Quadcopter*

3.4.1 Sensor *Gyroscope* dan *Accelerometer* [6]

Modul kontroler tersebut memiliki berbagai macam sensor untuk menunjang kelancaran dalam sistem, mulai dari *gyroscope, accelerometer,* dan barometer. Sensor *Accelerometer* adalah sensor yang digunakan untuk mengukur percepatan suatu objek. Accelometer mengukur percepatan dinamis dan statis. Pengukuran dinamis adalah pengukuran percepatan pada objek bergerak, sedangkan pengukuran statis adalah pengukuran terhadap gravitasi bumi. Sensor *gyroscope* merupakan perangkat untuk mengukur atau mempertahankan orientasi, dengan prinsip ketetapan momentum sudut.

IMU merupakan sensor yang berfungsi untuk menghitung percepatan serta orientasi arah pergerakan dari kendaraan udara dengan menggunakan kombinasi dari sensor *accelerometer* dan *gyroscope*. Dengan adanya IMU, kendaraan udara bisa menghitung dan mengetahui pergerakan yang dilakukannya, sehingga dapat membantu kendaraan tersebut untuk mengetahui posisi serta lintasan yang dilaluinya tanpa menggunakan GPS (misalkan ketika tidak mendapatkan sinyal GPS).

Secara umum IMU bekerja dengan menggunakan tiga sensor *accelerometer* yang digunakan untuk menghitung percepatan di sumbu x, y, dan z. Nantinya, *accelerometer* akan dipadukan dengan *gyroscope* untuk menentukan arah mana yang sedang diambil oleh *quadcopter* ketika melakukan percepatan tersebut. Dengan mencatat dan menggabungkan semua perhitungan tersebut, akan didapatkan posisi baru dari *quadcopter* yang bergerak, serta jalur pergerakan yang diambilnya. Sensor yang digunakan adalah MPU6050 seperti yang ditampilkan pada Gambar 3.7.

Spesifikasi dari sensor *accelerometer* sebagai berikut :

Batas operasi	: ±3,6 g,
Sensitivitas	: 300 mV/g,
Batas suplai tegangan	: 1,8 – 5 Volt.

Spesifikasi dari sensor *gyroscope* sebagai berikut :

Batas operasi	: ± 500 ^{0}/s,
Sensitivitas	: 2,0 mV/^{0}/s,
Batas supplai tegangan	: 3 Volt.

Gambar 3.7 Sensor MPU6050

Pada penilitan tugas akhir ini, sensor barometer tidak digunakan dan nilainya dihiraukan. Hal ini dikarenakan fungsi dari barometer untuk mengukur tekanan udara dan memberikan nilai ketinggan dari *quadrotor.*

3.4.2 Mikrokontroler ATmega 2560 [6]

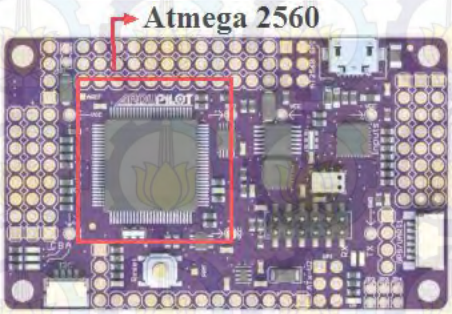
Mikrokontroler adalah piranti elektronik berupa *Integrated Circuit* (IC) yang memiliki kemampuan manipulasi data (informasi) berdasarkan suatu urutan instruksi (program) yang dibuat oleh programmer dimana di dalamnya sudah terdapat *Central Proccesssing Unit* (CPU), *Random Acess Memory* (RAM), *Electrically Erasable Programmable Read Only Memory* (EEPROM) , I/O, *Timer* dan peralatan internal lainnya yang sudah saling terhubung terorganisasi dengan baik oleh pabrik pembuatnya dan dikemas dalam satu chip yang siap pakai.

Umumnya mikrokontroler memiliki instruksi manipulasi bit, akses ke I/O secara langsung serta proses interupsi yang cepat dan efisien. Mikrokontroler yang digunakan adalah mikrokontroler ATMega 2560. Mikrokontroler akan diprogram menggunakan bahasa C. Mikrokontroler akan dipadukan menjadi modul *ardupilotmega* seperti pada Gambar 3.8

Spesifikasi dari mikrokontroler ATMega 2560 adalah:

Memori	: 256 KByte
Frekuensi *Clock*	: Maksimum 16Mhz
I/O	: 86 *Port*
ADC	: 10 bit/16 *port*
Timer 8 bit	: 2

Timer 16 Bit : 4
Supplai tegangan : 4,5 - 5,5 VDC



Gambar 3.8 Mikrokontroler ATmega 2560

3.4.3 *Transmitter* dan *Receiver Remote Control* [7]

Transmitter dan *receiver* digunakan untuk operasi manual melakukan manuver pada *quadrotor*. Hal ini dilakukan menggunakan *Remote Control* (*Analog*) yang sudah di kalibrasi sebelumnya sesuai dengan desain quadcopter yang telah di buat. *Transmitter* berada pada sisi pengguna untuk memberikan nilai *input* kepada *quadrotor*, sinyal tersebut akan diterima *receiver* dan diteruskan ke mikrokontroler berupa pulsa. Nantinya, mikrokontroller akan memberikan perintah terhadap motor untuk melakukan gerak elektrik sesuai perintah. Tampilan *transmitter* dan *receiver* radio seperti pada Gambar 3.9.

Spesifikasi dari radio *transmiter* adalah:

Nama *Transmitter* : Turnigy 9xR
Channel : 9
Modulasi : 2,4 GHz,
Baterai : 2500 mAh Li-Po – 11,1 V

Sedangkan spesifikasi untuk *receiver* adalah :

Nama *receiver* : Fr SKY V8FR *receiver*,
Ukuran modul utama : 45mm x 26mm x 17mm,
Ukuran modul satelit : 17mm x 23mm x 5mm,
Berat : 15 gram,
Range tegangan : 3,2 - 9,6 V,
Arus : 20 mA,


Gambar 3.9 *Remote Control Transmiter* dan *Receiver*

3.4.4 Modul *Electronic Speed Controller* (ESC) [8]

ESC atau disebut juga *Electronic Speed Control* adalah *driver* penggerak untuk jenis *motor brushless*, biasanya digunakan pada bidang *aeronautical* atau RC. Untuk melakukan *interface* dengan ESC, caranya cukup mudah, yaitu dengan memberikan pulsa pada pin input ESC yang akan berpengaruh pada kecepatan motor brushless. Alat ini merupakan driver dari motor *brushless* DC. ESC di-*trigger* oleh sinyal PWM yang dikendalikan oleh mikrokontroler ATMega 2560. Sinyal PWM tersebut akan *mendrive* motor dengan kecepatan yang linier dengan besar pulsa yang diberikan.

Jenis ESC yang digunakan adalah *TBS* Bulletproof 30A yang dapat dialiri arus hingga 30 *Ampere*, seperti pada Gambar 3.10. Spesifikasi ESC TBS Bulletproof 30A adalah:

Arus Maksimal : 30 Ampere,
Voltage : 6,4 – 16,8 Volt

Gambar 3.10 *Electronic Speed Controller*

3.4.5 MOTOR BLDC Sunny Sky V2216 900 kV [9]

Motor *brushless* merupakan perangkat yang digunakan untuk memutar p*ropeller quadcopter*. Jenis motor ini menggunakan sumber listrik DC sebagai sumber energi. Motor *brushless* ini sebenarnya merupakan motor AC tiga fasa, di mana putaran pada motor disebabkan oleh medan magnet pada stator yang pada setiap saatnya hanya aktif dua fasa (hanya dua fasa yang ter-*supply* pada setiap saat sementara satu fasa lainnya tidak ter-*supply*). Motor *brushless* yang digunakan sebagai aktuator pada penelitian tugas akhir ini adalah motor DC tanpa sikat Sunny Sky V2216 900 kV yang sumbunya dipasang propeler berukuran 10 cm x 4,5 cm. Bentuk motor DC tanpa sikat dan propeler terlihat seperti pada Gambar 3.11.

Gambar 3.11 Motor Brushless DC dan *Propeller*

Motor *brushless* yang digunakan pada perancangan sistem quadcopter ini adalah motor *brushless* merk SunnySky X2216 KV900 (Gambar 3.6) dengan spesifikasi sebagai berikut.

RPM/Volt : 900 KV,
Berat : 56 gram,
Shaft : 5 mm,
Propeller : 8,0x4,5 – 11,0x4,5,
Gaya angkat : 0,8 kg dengan menggunakan *propeller* 10,0x45
ESC yang dibutuhkan : 30A.

3.4.6 APM Planner 2.6 (*Flight Controller*) [10]

APM merupakan Ardupilot Mega yang didalamnya sudah tertanam berbagai macam komponen yang diperlukan untuk menerbangkan pesawat UAV. Selain itu, dalam APM ini sudah berisi program dasar dan *library quadcopter* yang nantinya secara otomatis dapat di hubungkan dengan software Mission Planner atau APM Planner dengan melakukan kalibrasi sensor terlebih dahulu. APM Planner merupakan otak inti dari *quadcopter*, karena melalui APM ini semua komponen elektrik dikendalikan.

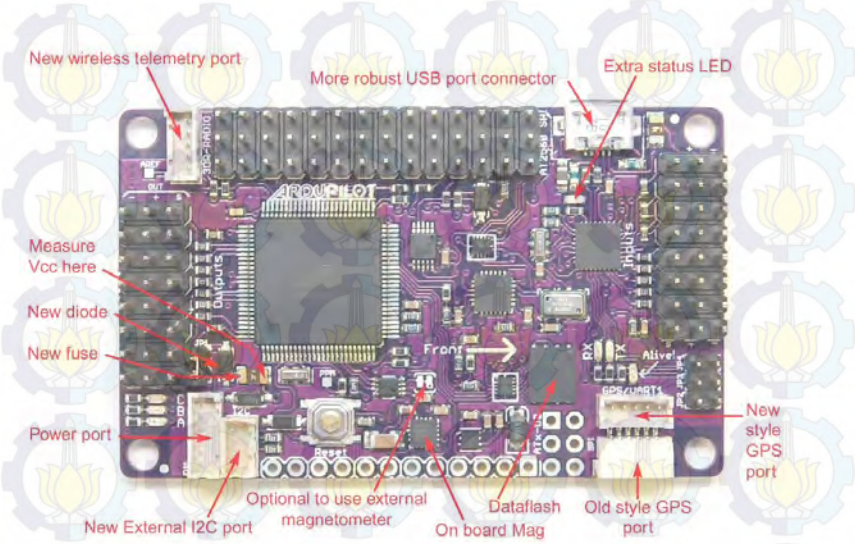
Jenis APM yang digunakan dalam penelitian tugas akhir ini adalah APM Planner 2.6. Didalam APM ini sudah tertanam Atmega 2560 sebagai mikrokontrolernya. Gambar 3.12 adalah tampilan dari APM yang digunakan.

Gambar 3.12 APM Planner 2.6

Spesifikasi dari APM Planner 2.6 sebagai berikut :

Prosesor	: Atmega 2560
Sensor (Dalam *Board*)	: 3-*axis gyroscope*, 3-*axis accelerometer*, *barometer*, dan *optional magnitude*
Data *Logging Memory*	: 2 Mega Byte
Ukuran	: 40 x 72 x 20 mm
Assembly Required	: *Some soldering*

Konfigurasi *input, output*, dan komponen lainnya dari APM Planner 2.6 akan di tampilkan pada Gambar 3.13.



Gambar 3.13 Konfigurasi APM Planner 2.6

3.4.7 Baterai Readymaderc 5100 mAh 3S 35C Lipo Pack [11]

Baterai Lithium Polimer atau biasa disebut dengan LiPo merupakan salah satu jenis baterai yang sering digunakan dalam dunia RC (*Remote Control*). Utamanya untuk RC tipe pesawat dan helikopter.

Ada tiga kelebihan utama yang ditawarkan oleh baterai berjenis LiPo daripada baterai jenis lain seperti NiCad atau NiMH yaitu :

1. Baterai LiPo memiliki bobot yang ringan dan tersedia dalam berbagai macam bentuk dan ukuran. Selain itu, baterai Lipo banyak tersedia di pasaran.
2. Baterai LiPo memiliki kapasitas penyimpanan energi listrik yang besar
3. Baterai LiPo memiliki tingkat discharge rate energi yang tinggi, dimana hal ini sangat berguna sekali dalam bidang RC.

Jenis baterai LiPo yang digunakan dalam tugas akhir ini adalah Readymaderc 5100mAh 3S 35C Lipo Pack yang akan di tampilkan pada Gambar 3.14 dengan spesifikasi sebagai berikut :

Kapasitas : 5100mAh
Tipe : 3S1P / 11,1v / 3Cell Lithium Polymer
Pengosongan Konstan : 35C
Berat : 380,2 g (termasuk konektor)
Ukuran : 138 x 43,7 x 28,9 mm
Charge Plug : JST-XH
Konektor Tegangan : T Connector
Arus Pengisian : Disarankan1-3C , 5C Max (1C = 5,1 Amps *charging rate*)
Watt - *hours* : 56,61
Power Wire : 10 AWG, 100 mm
Balance Wire : 22 AWG, 80 mm

Gambar 3.14 Baterai LiPo 5100 mAh

3.5 Pemodelan *Quadcopter* [12]

Persamaan model dari *quadcopter* didapatkan melalui pemodelan fisik. Dalam perancangan simulasi *quadcopter* ada 12 buah keluaran yang nantinya menentukan gerak-gerak dari *quadcopter*. Persamaan tersebut dituliskan pada Persamaan 3.1-3.6.

$$\ddot{x} = (sin\psi sin\phi + cos\psi sin\theta cos\phi)\frac{U1}{m} \quad (3.1)$$

$$\ddot{y} = (-cos\psi sin\phi + sin\psi sin\theta cos\phi)\frac{U1}{m} \quad (3.2)$$

$$\ddot{z} = -g + (cos\theta cos\phi)\frac{U1}{m} \quad (3.3)$$

$$\dot{p} = \frac{Iyy-Izz}{Ixx}qr + \frac{Jr}{Ixx}q\Omega + \frac{U2}{Ixx} \quad (3.4)$$

$$\dot{q} = \frac{Izz - Ixx}{Iyy} pr + \frac{Jr}{Iyy} p\Omega + \frac{U3}{Iyy} \quad (3.5)$$

$$\dot{r} = \frac{Ixx - Iyy}{Izz} pq + \frac{U4}{Izz} \quad (3.6)$$

3.5.1 Model Kinematika dan Dinamika *Quadcopter*

Persamaan 3.1 sampai 3.6 didapatkan dari hasil penurunan rumus yang akan dijelaskan sebagai berikut :

3.5.1.1 Model Kinematika *Quadcopter*

Persamaan 3.7 menunjukkan kinematika dari 6-*DOF* rigid-body

$$\dot{\xi} = J_\Theta v \quad (3.7)$$

$\dot{\xi}$ adalah vektor kecepatan yang mengacu pada *E-frame*, ν adalah vektor kecepatan mengacu *B-frame* dan J_Θ adalah matrik *jacobian*.

ξ terdiri dari *quadcopter linier* Γ^E [m] dan sudut Θ^E [rad] posisi pada *E-frame* seperti terlihat pada Persamaan (3.8).

$$\xi = [\Gamma^E \quad \Theta^E]^T = [X \quad Y \quad Z \quad \phi \quad \theta \quad \psi]^T \quad (3.8)$$

demikian pula dengan ν, terdiri dari *quadcopter linier* V^B [ms-1] dan kecepatan sudut ω^B [rad s-1] pada *B-frame* seperti yang ditunjukkan dalam Persamaan 3.9

$$v = [V^B \; \omega^B]^T = [u \quad v \quad w \quad p \quad q \quad r]^T \quad (3.9)$$

Matrik *jacobian* terdiri dari 4 *sub*-matrik sebagaimana Persamaan 3.10

$$J_\Theta = \begin{bmatrix} R_\Theta & 0_{3x3} \\ 0_{3x3} & T_\Theta \end{bmatrix} \quad (3.10)$$

Matrik rotasi (R_Θ) dan matrik translasi (T_Θ) ditunjukkan pada Persamaan 3.11 dan 3.12

$$R_\Theta = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - c\phi\psi & c\psi s\theta c\phi + s\psi s\phi \\ s\phi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\theta c\phi \end{bmatrix} \quad (3.11)$$

$$T_\Theta = \begin{bmatrix} 1 & tan\theta sin\phi & tan\theta cos\phi \\ 0 & cos\phi & -sin\phi \\ 0 & sec\theta sin\phi & sec\theta cos\phi \end{bmatrix} \quad (3.12)$$

Variabel *x*, *y*, dan *z* adalah posisi pada *frame* bumi, sedangkan kecepatan *u*, *v*, dan *w* pada *frame* badan *quadcopter*. Hubungan antara posisi dan kecepatan ini terlihat pada Persamaan 3.13 dan 3.14.

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R\begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (3.13)$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - c\phi\psi & c\psi s\theta c\phi + s\psi s\phi \\ s\phi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\theta c\phi \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (3.14)$$

3.5.1.2 Model Dinamika *Quadcopter*

Dinamika *rigid body* 6 *DOF* memperhitungkan massa *body* m [kg] dan matrik inersia I [N m s2]. Dinamika diperlihatkan oleh Persamaan 3.15

$$\begin{bmatrix} m\,I_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix}\begin{bmatrix} \dot{V}^B \\ \dot{\omega}^B \end{bmatrix} + \begin{bmatrix} \omega^B x\,(m\,V^B) \\ \omega^B x\,(I\,\omega^B) \end{bmatrix} = \begin{bmatrix} F^B \\ \tau^B \end{bmatrix} \quad (3.15)$$

Notasi $I_{3\times3}$ berarti matrik identitas 3 kali 3. $\dot{V}^B$ [ms-2] adalah vector linier percepatan *quadcopter* mengacu *B-frame* sementara $\dot{\omega}^B$ [rad s-2] adalah vektor percepatan sudut *quadcopter* mengacu *B-frame*. Selain itu, F^B [N] adalah vektor gaya *quadcopter* mengacu *B-frame* dan τ^B [N m] adalah vektor torsi *quadcopter* pada *B-frame*.

Hukum Newton kedua tentang gerakan berlaku rumus

$$F = m\frac{dv}{dt_e} \quad (3.16)$$

m adalah massa dari sistem yang digunakan, *a* adalah percepatan gerak, dan *F* adalah gaya yang terjadi pada sistem. Dari Persamaan *Coriolis* didapatkan Persamaan 3.17 dan Persamaan 3.18

$$F = m\frac{dv}{dt_e} = m\left(\frac{dv}{dt_b} + \omega \times v\right) \quad (3.17)$$

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m\frac{dv}{dt_e} = m\left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix}\right) \quad (3.18)$$

Sehingga percepatan linear dari *quadcopter* dapat dihitung :

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} - \begin{bmatrix} qw - rv \\ ru - vw \\ pv - qu \end{bmatrix} \quad (3.19)$$

Untuk menghitung gaya putar yang terjadi pada *quadcopter* menggunakan hukum Newton kedua seperti pada Persamaan 3.20

$$M = \frac{dh}{dt_e} \quad (3.20)$$

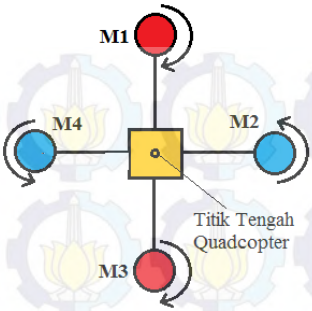
M adalah *moment* yang terjadi, dan h merupakan momentum putar. Dengan menggunakan persamaan *Coriolis* didapatkan :

$$M = \frac{dv}{dt_e} = \frac{dh}{dt_b} + \omega \times h \quad (3.21)$$

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = I\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I\begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (3.22)$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = I^{-1}\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - I^{-1}\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I\begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (3.23)$$

I merupakan matrik dari momen inersia yang terjadi pada sumbu Xb, Yb, dan Zb yang direpresentasikan kedalam variabel *Ixx*, *Iyy*, dan *Izz*. Perhitungan momen inersia pada *quadcopter* ada 2 bagian, yaitu *Ixx*=*Iyy* dan *Izz*.



Gambar 3.15. Momentum Inersia Pada Sumbu Xb, Yb dan Zb

Ada beberapa asumsi yang diberikan sebelum menghitung momen inersia terhadap sumbu Xb (dan Yb) dari *quadcopter*. Asumsi-asumsi tersebut adalah :

1. Motor M_1 dan M_3 berbentuk silindris dengan jari-jari ρ, tinggi h, dan massa m.
2. *Body* tengah dari *quadcopter* dianggap berbentuk kotak dengan jari-jari R, tinggi H, dan massa M.

Momentum inersia terhadap sumbu x ada 2 bagian, yaitu :

1. Hubungan gaya dari Motor M_2 dan M_4 terhadap sumbu x dengan jari-jari putaran l.
2. Hubungan Motor M_1, M_3, dan *body* tengah terhadap sumbu Xb

Momen inersia bentuk silindris yang tegak lurus terhadap badan *quadcopter* terlihat pada Persamaan 3.24.

$$I = \frac{massa(jari-jari)^2}{4} + \frac{massa(tinggi)^2}{12} \tag{3.24}$$

Bagian pertama, hubungan gaya dari motor M_2 dan M_4 terhadap sumbu Xb ditampilkan pada Persamaan 3.25. Bagian kedua, hubungan motor M_1, M_3, dan *body* tengah terhadap sumbu Xb ditampilkan pada Persamaan 3.26

$$Ixx_1 = 2ml^2 \tag{3.25}$$

$$Ixx_2 = 2\left[\frac{ml^2}{4} + \frac{mh^2}{12}\right] + \frac{MR^2}{4} + \frac{MH^2}{12} \tag{3.26}$$

Sehingga total persamaan Ixx dan Iyy atau penjumlahan Persamaan 3.27 dan 3.28 adalah

$$Ixx = \frac{m\rho^2}{4} + \frac{mh^2}{6} + 2ml^2 + \frac{MR^2}{4} + \frac{MH^2}{12} \quad (3.27)$$

$$Iyy = \frac{m\rho^2}{4} + \frac{mh^2}{6} + 2ml^2 + \frac{MR^2}{4} + \frac{MH^2}{12} \quad (3.28)$$

Untuk menghitung momentum inersia pada sumbu Zb, dapat dibagi menjadi 2 bagian, yaitu :

1. Momen inersia pada badan tengah *quadcopter*.
2. Hubungan motor M_1, M_2, M_3, dan M_4.

Bagian pertama, momentum inersia pada badan tengah *quadcopter* adalah terlihat pada Persamaan 3.29. Bagian kedua, hubungan motor M_1, M_2, M_3, dan M_4 terlihat pada Persamaan 3.30.

$$Izz_1 = \frac{MR^2}{2} \quad (3.29)$$

$$Izz_2 = 4ml^2 \quad (3.30)$$

Total Persamaan Izz adalah

$$Izz = \frac{MR^2}{2} + 4ml^2 \quad (3.31)$$

$$I = \begin{bmatrix} Ixx & 0 & 0 \\ 0 & Iyy & 0 \\ 0 & 0 & Izz \end{bmatrix}, \; I^{-1} = \begin{bmatrix} \frac{1}{Ixx} & 0 & 0 \\ 0 & \frac{1}{Iyy} & 0 \\ 0 & 0 & \frac{1}{Izz} \end{bmatrix} \quad (3.32)$$

Dengan memasukkan matrik I dan I^{-1} ke dalam Persamaan 3.23 maka akan didapatkan Persamaan 3.33 dan Persamaan 3.34

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{1}{Ixx} & 0 & 0 \\ 0 & \frac{1}{Iyy} & 0 \\ 0 & 0 & \frac{1}{Izz} \end{bmatrix} \left(\begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{bmatrix} Ixx & 0 & 0 \\ 0 & Iyy & 0 \\ 0 & 0 & Izz \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \begin{bmatrix} \tau\phi \\ \tau\theta \\ \tau\psi \end{bmatrix} \right) \quad (3.33)$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\frac{Iyy-Izz}{Ixx}qr\\ \frac{Izz-Iyy}{Iyy}pr\\ \frac{Ixx-Iyy}{Izz}pq\end{bmatrix} + \begin{bmatrix}\frac{1}{Ixx}\tau\phi\\ \frac{1}{Iyy}\tau\theta\\ \frac{1}{Izz}\tau\psi\end{bmatrix} \quad (3.34)$$

Dapat dituliskan lagi persamaan kinematik dan dinamik yang telah didapatkan sebelumnya

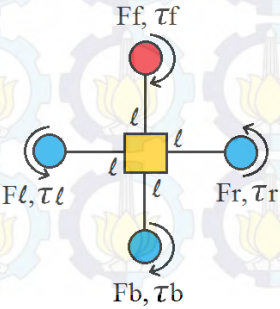
$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}c\psi c\theta & c\psi s\theta s\phi - c\phi\psi & c\psi s\theta c\phi + s\psi s\phi\\ s\phi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi\\ -s\theta & s\phi c\theta & c\theta c\phi\end{bmatrix}\begin{bmatrix}u\\ v\\ w\end{bmatrix} \quad (3.35)$$

$$\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} = \frac{1}{m}\begin{bmatrix}F_x\\ F_y\\ F_z\end{bmatrix} - \begin{bmatrix}qw - rv\\ ru - vw\\ pv - qu\end{bmatrix} \quad (3.36)$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\frac{Iyy-Izz}{Ixx}qr\\ \frac{Izz-Iyy}{Iyy}pr\\ \frac{Ixx-Iyy}{Izz}pq\end{bmatrix} + \begin{bmatrix}\frac{1}{Ixx}\tau\phi\\ \frac{1}{Iyy}\tau\theta\\ \frac{1}{Izz}\tau\psi\end{bmatrix} \quad (3.37)$$

3.5.1.3 Gaya dan Momen

Pada bagian ini dibahas mengenai gaya dan momen yang bekerja pada *quadcopter*. Pada *quadcopter* dianggap tidak ada yang berbentuk *aerodinamis* sehingga gaya *aerodinamis* dan momen *aerodinamis* yang terjadi dapat diabaikan.



Gambar 3.16 Gaya ***F*** dan Torsi $\boldsymbol{\tau}$ pada *Quadcopter*

Dari gaya-gaya yang terjadi pada tiap motor di *quadcopter*, dapat dihitung persamaan torsi yang terjadi pada *roll*, *pitch*, dan *yaw*.

$$U_1 = F = F_f + F_r + F_b + F_l = b\left(\Omega_f^2 + \Omega_r^2 + \Omega_b^2 + \Omega_l^2\right) \quad (3.38)$$

$$U_2 = \tau\phi = bl(-\Omega_r^2 + \Omega_l^2) \quad (3.39)$$

$$U_3 = \tau\theta = bl\left(\Omega_f^2 - \Omega_b^2\right) \quad (3.40)$$

$$U_4 = \tau\psi = d\left(-\Omega_f^2 + \Omega_r^2 - \Omega_b^2 + \Omega_l^2\right) \quad (3.41)$$

Atau dapat ditulis juga

$$\mathrm{U_B}(\Omega) = \mathrm{E}\Omega^2 = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ F \\ \tau\phi \\ \tau\theta \\ \tau\psi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b\left(\Omega_f^2 + \Omega_r^2 + \Omega_b^2 + \Omega_l^2\right) \\ bl(-\Omega_r^2 + \Omega_l^2) \\ bl\left(\Omega_f^2 - \Omega_b^2\right) \\ d\left(-\Omega_f^2 + \Omega_r^2 - \Omega_b^2 + \Omega_l^2\right) \end{bmatrix} \quad (3.42)$$

$\mathrm{U_B}$ ($\boldsymbol{\Omega}$) merupakan *movement vector, b* merupakan konstanta *thrust* dan d adalah konstanta *drag* yang terjadi pada *quadcopter*. Untuk mencari konstanta tersebut perlu dilakukan percobaan dan hubungan konstanta tersebut adalah proporsional terhadap kuadrat dari kecepatan motor yang akan dijelaskan nanti.

Selain gaya yang dihasilkan motor, gravitasi juga mempengaruhi gaya yang terjadi pada *quadcopter*. Persamaan gaya gravitasi terhadap *frame* bada *quadcopter* dapat dilihat pada Persamaan 3.43, sedangkan terhadap *frame* bumi dapat dilihat pada Persamaan 3.44.

$$f_g = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \quad (3.43)$$

$$F_g = R\begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} = \begin{bmatrix} mgsin\theta \\ -mgsin\phi cos\theta \\ -mgcos\theta cos\phi \end{bmatrix} \quad (3.44)$$

$$G_B(\xi) = \begin{bmatrix} F_G^B \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} R_\Theta^{-1} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} R_\Theta^T \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} mgsin\theta \\ -mgsin\phi cos\theta \\ -mgcos\theta cos\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.45)$$

Kontribusi selanjutnya memperhitungkan efek *gyroscopic* yang dihasilkan oleh rotasi baling-baling. Karena dua dari mereka yang berputar searah jarum jam dan dua lainnya berlawanan, ada ketidakseimbangan secara keseluruhan ketika jumlah aljabar dari kecepatan *rotor* tidak sama dengan nol. Selain itu jika kecepatan sudut *roll* atau *pitch* juga berbeda dari nol, *quadcopter* mengalami torsi *gyroscopic* menurut Persamaan 3.46

$$O_b(v)\Omega = \begin{bmatrix} -\sum_{k=1}^{4} \mathrm{Jtp} \left(\mathrm{w^b x} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} (-1)^k \Omega_k \right) \end{bmatrix} = \begin{bmatrix} \mathrm{O_{3x1}} \\ \mathrm{Jtp} \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \Omega \end{bmatrix}$$

$$= \mathrm{Jtp} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \Omega \quad (3.46)$$

$O_b(v)\Omega$ adalah *propeller gyroscopic* matrik dan JTP [N m s2] atau Jr [N m s2] Adalah momen inersia rotasi total sekitar sumbu *propeller* dihitung dengan Persamaan 3.47

$$JTP = \frac{M}{2} r^2 \quad (3.47)$$

Inersia rotasi total sekitar sumbu *propeller* terjadi disekitar motor, sehingga M adalah masa motor, sedangkan r adalah jari-jari motor.

Persamaan 3.48 mendefinisikan keseluruhan kecepatan *propeller* Ω [rad s-1] yang digunakan dalam Persamaan 3.46

$$\Omega = -\Omega_f + \Omega_r - \Omega_b + \Omega_l \quad (3.48)$$

Dengan adanya gaya grafitasi dan *gyroscopic* yang dihasilkan oleh rotasi baling-baling, maka model kinematik dan dinamik pada Persamaan 3.35-3.36 menjadi :

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}c\psi c\theta & c\psi s\theta s\phi - c\phi\psi & c\psi s\theta c\phi + s\psi s\phi\\ s\phi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi\\ -s\theta & s\phi c\theta & c\theta c\phi\end{bmatrix}\begin{bmatrix}u\\ v\\ w\end{bmatrix} \quad (3.49)$$

$$\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} = \frac{1}{m}\begin{bmatrix}0\\ 0\\ -U_1\end{bmatrix} - \begin{bmatrix}qw - rv\\ ru - vw\\ pv - qu\end{bmatrix} + \begin{bmatrix}gsin\theta\\ -gsin\phi cos\theta\\ -gcos\theta cos\phi\end{bmatrix} \quad (3.50)$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\frac{Iyy-Izz}{Ixx}qr\\ \frac{Izz-Iyy}{Iyy}pr\\ \frac{Ixx-Iyy}{Izz}pq\end{bmatrix} + \begin{bmatrix}\frac{1}{Ixx}\tau\phi\\ \frac{1}{Iyy}\tau\theta\\ \frac{1}{Izz}\tau\psi\end{bmatrix} \quad (3.51)$$

Sistem *quadcopter* dinamis dalam Persamaan 3.51-3.49 ditulis dalam kerangka tubuh-tetap. Sebagaimana dinyatakan sebelumnya, referensi ini banyak digunakan dalam 6-*DOF*. Namun dalam kasus ini dapat berguna untuk mengungkapkan dinamika yang berkaitan dengan sistem *hibrida* yang terdiri dari Persamaan linear *E-frame* dan Persamaan sudut *B-frame*. Oleh karena itu Persamaan 3.52 akan dinyatakan dalam bingkai baru "*hybrid*" yang disebut *H-frame*. Referensi baru ini diadopsi karena dapat mempermudah dalam mengungkapkan dinamika yang akan dikombinasikan dengan kontrol (khususnya untuk posisi vertikal dalam kerangka inersial bumi). Persamaan 2.46 menunjukkan vektor kecepatan *quadcopter* mengacu *H-frame* (ζ).

$$\xi = [\dot{\Gamma}^E \quad \omega^B]^T = [\dot{X} \quad \dot{Y} \quad \dot{Z} \quad p \quad q \quad r]^T \quad (3.52)$$

Dinamika sistem pada *H-frame* dapat dituliskan dengan matrik sebagaimana Persamaan 3.53

$$M_H\dot{\xi} + C_H(\xi)\xi = G_H + O_H(\xi)\Omega + E_H(\xi)\Omega^2 \quad (3.53)$$

Dimana ξ adalah vector akselerasi *quadcopter* yang terjadi pada *H-frame*. Sedangkan M_H adalah matrik inersia sistem pada *H-frame* yang sesuai dengan *B-frame*

$$M_H = M_B = \begin{bmatrix} m\,I_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xx} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{yy} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{zz} \end{bmatrix} \quad (3.54)$$

Sedangkan, matrik *Coriolis-sentripetal* adalah

$$C_H = \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & -S(I\omega^B) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{zz}r & -I_{yy}q \\ 0 & 0 & 0 & -I_{zz}r & 0 & I_{xx}p \\ 0 & 0 & 0 & I_{yy}q & -I_{xx}p & 0 \end{bmatrix} \quad (3.55)$$

Matrik grafitasi G menjadi

$$G_H = \begin{bmatrix} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.56)$$

Untuk efek *gyroscopic* yang terjadi pada *H-frame* sama dengan *B-frame*, sehingga matrik O_H=O_B

Untuk matrik gerakan pada *H-frame* E_H (ξ) berbeda dengan pada *B-frame* karena masukan U_1 mempengaruhi kesemua dari tiga Persamaan linier melalui rotasi matrik R_θ. Perkalian produk antara matrik gerakan dan vektor kecepatan kuadrat gerakan baling-baling ditampilkan dalam Persamaan 3.57

$$E_H(\xi)\,\Omega^2 = \begin{bmatrix} R_\Theta & 0_{3x3} \\ 0_{3x3} & I_{3x3} \end{bmatrix} E_B\Omega^2 = \begin{bmatrix} (s_\psi s_\phi + c_\psi s_\theta c_\phi)U_1 \\ (-c_\psi s_\phi + s_\psi s_\theta c_\phi)U_1 \\ (s_\theta c_\phi)U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \quad (3.57)$$

Dengan membalik Persamaan 2.47 adalah mungkin untuk mengisolasi turunan dari kecepatan vektor pada *H-frame* menjadi

$$\dot{\xi} = {M_H}^{-1}(-C_H(\xi)\xi + G_H + O_H(\xi)\Omega + E_H(\xi)\Omega^2) \quad (3.58)$$

3.5.1.4 Konstanta *Thrust* dan *Drag*

Konstanta *drag* dihitung dengan Persamaan gerak lurus berubah beraturan. Gerak lurus berubah beraturan adalah gerak yang lintasannya berupa garis lurus dengan kecepatannya yang berubah beraturan. Jadi pengukuran konstanta drag dilakukan dengan mengambil data terbang *quadcopter* pada saat *quadcopter take-off*. Gaya yang terjadi saat *quadcopter* bergerak keatas adalah

$$\sum F = Thrust - gravitasi - Drag \quad (3.59)$$

$$ma = (\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg - D \quad (3.60)$$

Sehingga

$$D = (\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg - ma \quad (3.61)$$

Dimana

$$a = \frac{s - v_o * t}{0{,}5 * t^2} \quad (3.62)$$

dan

$$CD = \frac{D}{\Omega^2} \quad (3.63)$$

Dengan ketentuan :

CD = Konstanta *drag*
v_o = Kecepatan awal (m/detik)
t = Waktu (detik)
a = Percepatan (m/detik^2)
s = Jarak yang ditempuh (m)
m = Massa *quadcopter* (kg)
τ_i = Gaya angkat (N/m), i=1,2,3,4
D = Gaya drag

Sedangkan Konstanta *thrust* (CT atau b) dihitung dengan menggunakan Persamaan 3.64

$$CT = \frac{\tau}{\Omega^2} \quad (3.64)$$

3.5.1.5 Model Matematika

Dari analisis kinematika dan dinamika yang telah dibahas, diperoleh persamaan model matematika dari *quadcopter* secara keseluruhan di tunjukkan pada persamaan 2.65.

$$
\begin{cases}
\ddot{X} = \dfrac{U_1}{m}(\cos\Psi \sin\theta\cos\phi + \sin\Psi\sin\phi) \\
\ddot{Y} = \dfrac{U_1}{m}(\sin\Psi \sin\theta\cos\phi - \cos\Psi\sin\phi) \\
\ddot{Z} = -g + \dfrac{U_1}{m}(\cos\theta\cos\phi) \\
\dot{p} = \dfrac{I_{yy} - I_{zz}}{I_{xx}} qr - \dfrac{I_r}{I_{xx}} q\Omega + \dfrac{U_2}{I_{xx}} \\
\dot{q} = \dfrac{I_{zz} - I_{xx}}{I_{yy}} pr + \dfrac{I_r}{I_{yy}} q\Omega + \dfrac{U_2}{I_{yy}} \\
\dot{r} = \dfrac{I_{xx} - I_{yy}}{I_{zz}} pq + \dfrac{U_4}{I_{zz}}
\end{cases}
\tag{3.65}
$$

di mana,

$\ddot{X}$ = percepatan terhadap sumbu x
$\ddot{Y}$ = percepatan terhadap sumbu y
$\ddot{Z}$ = percepatan terhadap sumbu z
U_1 = gaya angkat/*thrust*
U_2 = gaya untuk melakukan *roll*
U_3 = gaya untuk melakukan *pitch*
U_4 = gaya untuk melakukan *yaw*
Φ = sudut *roll*
Θ = sudut *pitch*
ψ = sudut *yaw*
$\dot{p}$ = percepatan sudut terhadap sudut X
$\dot{q}$ = percepatan sudut terhadap sudut Y
$\dot{r}$ = percepatan sudut terhadap sudut Z
m = massa total *quadrotor*
I_{xx} = momen inersia *frame* pada sumbu X
I_{yy} = momen inersia *frame* pada sumbu Y
I_{zz} = momen inersia *frame* pada sumbu Z
I_r = momen inersia total pada sumbu *propeller*

Dengan melihat secara sederhana pada Persamaan 3.65, posisi pada sumbu Z, dan posisi sudut *roll*, *pitch*, *yaw* dapat dikontrol secara langsung, berturut-turut dengan menggunakan U_1, U_2, U_3,dan U_4.

Kontrol pada posisi maju (X), dan menyamping (Y) dapat dilakukan dengan mengatur sudut *pitch* dan (-*roll*) dengan syarat gaya angkat (U_1) tidak sama dengan nol.

Nilai *input* dari *quadcopter* merupakan gaya angkat tiap propeler yang dimodelkan secara teoritis adalah sebagai berikut:

$$\begin{cases} U_1 = b(\Omega_1^{\ 2} + \Omega_2^{\ 2} + \Omega_3^{\ 2} + \Omega_4^{\ 2}) \\ U_2 = bl(-\Omega_2^{\ 2} + \Omega_4^{\ 2}) \\ U_3 = bl(-\Omega_1^{\ 2} + \Omega_3^{\ 2}) \\ U_4 = d(-\Omega_1^{\ 2} + \Omega_2^{\ 2} - \Omega_3^{\ 2} + \Omega_4^{\ 2}) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \quad (3.66)$$

di mana,

Ω_1 = kecepatan putar motor ke-1
Ω_2 = kecepatan putar motor ke-2
Ω_3 = kecepatan putar motor ke-3
Ω_4 = kecepatan putar motor ke-4
Ω = kecepatan putar total
b = konstanta *thrust*
l = lebar *frame*
d = konstanta *drag* dari *quadrotor*

3.5.2 Persamaan Motor dan *Propeller*

Untuk mendapatkan model linier dari motor *quadcopter*, maka dibutuhkan dua uji coba. Percobaan ini dilakukan untuk mendapatkan data kecepatan motor (rad/s) dan menentukan besarnya gaya angkat motor (gram).

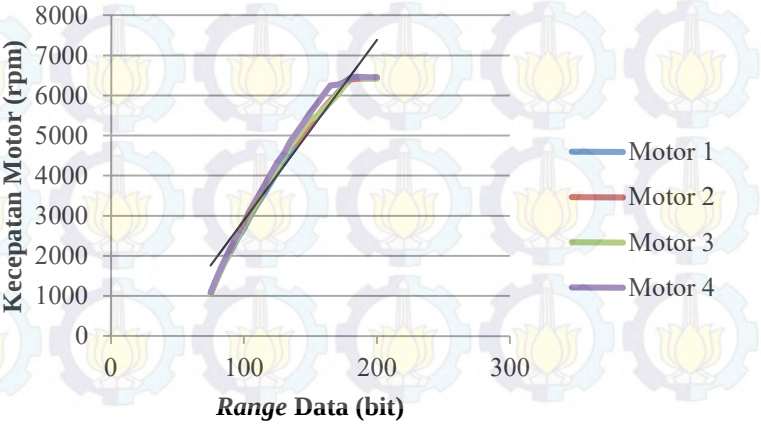
1. Percobaan 1

   Pada percobaan ini dilakukan pengukuran kecepatan motor menggunakan PWM (*pulse with modulation*) yang mengubah lebar pulsa (*duty cylce*) dengan nilai amplitudo dan frekuensi yang tetap, kemudian dihubungkan dengan arduino. Hal ini bertujuan untuk mengetahui hubungan lebar pulsa (*duty cycle)* dengan kecepatan motor. Fitur PWM pada arduino memiliki resolusi sebesar 8 bit jadi bernilai $2^8 = 256$, dengan range 0 - 255. Resolusi yang dimaksud yaitu rentang data (*range*) yang mampu dibaca oleh mikrokontroler terhadap nilai PWM nya.Untuk mengetahui nilai kecepatan motor maka digunakan

alat *tachometer*. Percobaan 1 akan di tampilkan pada Gambar 3.17

Gambar 3.17 Percobaan 1

Hasil pengukuran kecepatan motor *brushless* menggunakan metode modulasi PWM (*pulse with modulation*) ditunjukkan pada Gambar 3.18



Gambar 3.18 Hubungan Kecepatan Motor Dengan Lebar Pulsa

Dari pengukuran diatas diperoleh persamaan linier hubungan pulsa motor terhadap kecepatan (*rad/s*) pada masing-masing motor adalah :

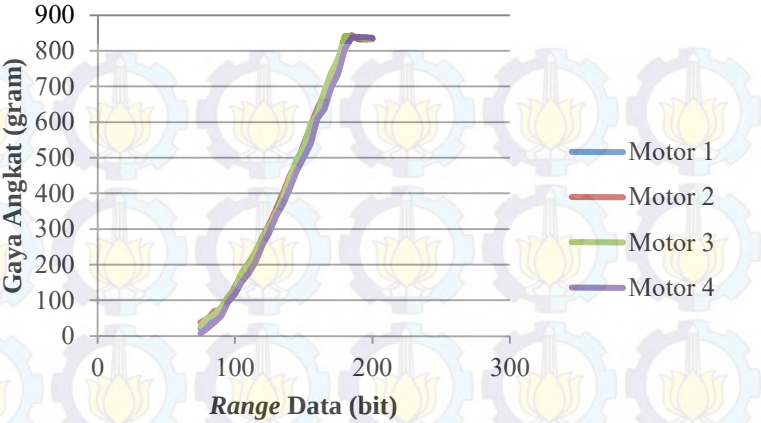
$$y = 44{,}851x - 1760{,}5 \quad (3.67)$$
$$y = 44{,}104x - 1603{,}7 \quad (3.68)$$
$$y = 44{,}58x - 1670{,}4 \quad (3.69)$$
$$y = 45{,}047x - 1618{,}7 \quad (3.70)$$

2. Percobaan 2

Pada percobaan ini dilakukan pengukuran gaya angkat (*thrust*) masing-masing motor *propeller* dengan cara yang ditampilkan pada Gambar 3.19

Gambar 3.19 Hubungan Lebar Pulsa Terhadap Gaya *Thrust* Motor

Langkah pengukurannya adalah dengan membuat suatu benda yang memiliki beban 1,2 kg (botol berisi air). Kemudian motor dan baling-baling dilekatkan pada ujung botol seperti terlihat pada Gambar 3.19. Motor berputar dengan memberikan *range* data 0 - 255 (bit) yang telah di program pada arduino. Hasil pembacaan berat pada timbangan dikurangi dengan beban 1,2 kg adalah gaya angkat pada masing-masing motor. Percobaan ini dilakukan pada masing- masing motor. Hasil pengukuran gaya angkat (thrust) motor terhadap pulsa PWM yang diberikan di perlihatkan pada Gambar 3.20.



Gambar 3.20 Hubungan Pulsa Terhadap Gaya Angkat Motor

Dari pengukuran yang telah dilakukan, maka diperoleh persamaan linier hubungan pulsa motor terhadap gaya angkat (gram) pada masing-masing motor adalah

$y = 7{,}6115x - 609{,}73$ (3.71)

$y = 7{,}5622x - 597{,}22$ (3.72)

$y = 7{,}5869x - 601{,}94$ (3.73)

$y = 7{,}6392x - 632{,}12$ (3.74)

3.5.3 Konstanta Inersia (Ixx, Iyy, dan Izz)

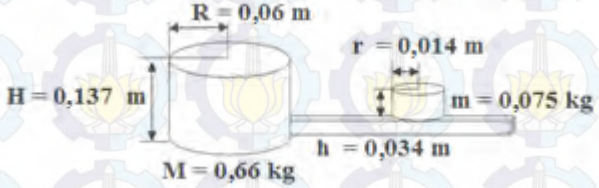
Momen inersia adalah ukuran kelembaman suatu benda untuk berotasi terhadap porosnya. Momen inersia berperan dalam dinamika rotasi seperti massa dalam dinamika dasar, dan menentukan hubungan antara momentum sudut dan kecepatan sudut, momen gaya dan percepatan sudut, dan beberapa besaran lain. Sesuai dengan dasar teori yang telah di jelaskan sebelumnya, maka dapat dihitung nilai parameter Ixx, Iyy, dan Izz dengan mengasumsi dari bentuk *quadcopter*. Persamaan yang digunakan untuk menghitung Ixx, Iyy, dan Izz akan di tunjukkan pada persamaan di bawah ini.

$$Ixx = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12} \quad (3.75)$$

$$Iyy = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12} \quad (3.76)$$

$$Izz = \frac{MR^2}{2} + 4mr^2 \quad (3.77)$$

Untuk medapatkan nilai Ixx, Iyy, dan Izz, maka harus diketahui dahulu massa, tinggi, dan jari-jari pada *quadcopter*. Untuk lebih jelasnya akan ditampilkan pada Gambar 3.21.



Gambar 3.21 *Massa*, Tinggi, dan Jari-jari *Quadcopter*

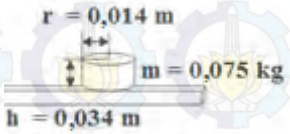
Dengan memasukkan nilai massa, tinggi, dan jari-jari sesuai gambar diatas ke dalam persamaan Ixx, Iyy, dan Izz, maka bisa di dapatkan nilai parameter Ixx adalah 1,680157 x 10^{-3} Kgm^2, Iyy adalah 1,680157 x 10^{-3} Kgm^2, dan Izz adalah 1,20466 x 10^{-3} Kgm^2

3.5.4 Konstanta Inersia Motor

Untuk mengetahui nilai momen inersia pada motor *quadcopter*, maka digunakan rumus pada persamaan 3.78.

$$Jr = \frac{4 \times (m \times r^2)}{2} \quad (3.78)$$

Panjang, *massa*, dan jari-jari dari motor yang digunakan pada *quadcopter* di tunjukkan pada Gambar 3.22.



Gambar 3.22 Panjang, *Massa*, dan Jari-jari Motor

Dengan memasukkan nilai panjang, massa, dan jari-jari motor sesuai gambar diatas ke dalam Persamaan 3.18, maka di dapatkan nilai dari momen inersia motor adalah 0,025872 Kgm^2

3.6 Identifikasi Konstanta

Dari model matematik yang diperoleh dengan pemodelan fisis, maka konstanta yang harus diperoleh untuk menentukan parameter *plant quadcopter* akan ditampilkan pada Tabel 3.1.

Tabel 3.1 Identifikasi Konstanta

Konstanta	Nilai	Satuan
Massa *Quadcopter*	1,2	Kg
Momen inersia rotasi sumbu X (Ixx)	1,680157 x 10^{-3}	Kgm^2
Momen inersia rotasi sumbu Y (Iyy)	1,680157 x 10^{-3}	Kgm^2
Momen inersia rotasi sumbu Z (Izz)	1,20466 x 10^{-3}	Kgm^2
Momen Inersia motor-propeler (JTP)	0,025872	Kgm^2
Konstanta *Thrust*	2,2478x10^{-6}	$N.sec^2$
Konstanta Drag	2,5x10^{-7}	$Nm.sec^2$

3.7 Model Matematis Dengan Konstanta

Setelah diperoleh konstanta-konstanta dari sistem, maka dapat dituliskan kembali model matematika dari sistem *quadcopter* adalah sebagai berikut:

$$\ddot{x} = (sin\psi sin\phi + cos\psi sin\theta cos\phi)\frac{U1}{1,2} \quad (3.79)$$

$$\ddot{y} = (-cos\psi sin\phi + sin\psi sin\theta cos\phi)\frac{U1}{1,2} \quad (3.80)$$

$$\ddot{z} = -9{,}81 + (cos\theta cos\phi)\frac{U1}{1,2} \quad (3.81)$$

$$\dot{p} = -0{,}5495qr + 0.0017q\Omega + 0.2052 \quad (3.82)$$

$$\dot{q} = 0.1675pr - 0.0094p\Omega + 2.955\ U3 \quad (3.83)$$

$$\dot{r} = -2{,}0257pq + 0.0594\ U4 \quad (3.84)$$

3.8 Perancangan Kontroler Pada Simulasi

Setelah diperoleh model matematika dan konstanta maka dapat dilakukan perancangan kontroler dengan mencari parameter-parameter kontroler yang diperlukan.

3.8.1 Kontroler PID (*Proportional-Integral-Derivatife Controller*) Untuk Gerak Rotasi

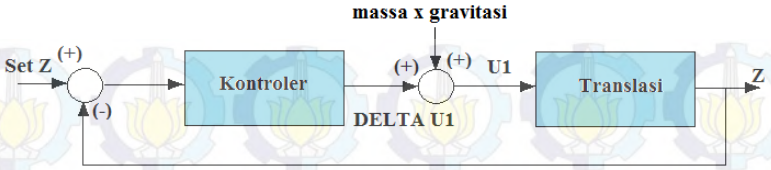
Kontroler PID merupakan kontroler untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tesebut. Model yang dirancang pada sistem *quadrotor* memiliki 12 keluaran yaitu 3 kecepatan translasi, 3 posisi linier, 3 kecepatan rotasi, dan terakhir 3 posisi sudut. Keluaran inilah yang digunakan sebagai umpan balik yang menuju kontroler. Diharapkan kontroler dapat melakukan koreksi terhadap kesalahan.

Quadcopter dapat melakukan sistem navigasi *waypoint* dengan cara mempertahankan posisi gerak *lateral.* Dalam mempertahankan posisi sudut *roll* dan *pitch*, serta posisi koordinat *x* dan *y,* harus dijaga kestabilannya sehingga keluaran 12 keluaran dalam sistem tidak digunakan seluruhnya. Dengan mengkategorikan persamaan percepatan X, Y dan Z adalah persamaan translasi serta persamaan Ø, Θ, dan ψ adalah rotasi. Bentuk umum persamaan matematis dari kontroler PID ditunjukkan pada Persamaan 3.7. Sedangkan kontroler PID dalam bentuk diskrit ditunjukkan pada Persamaan 3.8.

$$u(t) = K_p \left\{ e(t) + \frac{1}{\tau_i} \int_0^t e(t)dt + \tau_d \frac{de(t)}{dt} \right\} \tag{3.85}$$

$$u_{(k)} = K_p e_k + K_I T \sum_0^K e_k + \frac{1}{T} K_d \left(e_k - e_{k-1} \right) \tag{3.86}$$

Dengan mengkategorikan persamaan percepatan X, Y, dan Z adalah persamaan translasi, serta persamaan Ø, Θ, dan ψ adalah persamaan rotasi. Maka dapat dilihat bahwa untuk mengendalikan ketinggian dari *quadrotor*, diperoleh parameter yang akan dikendalikan adalah gaya *thrust* pada persamaan translasi. Sinyal kontrol yang diberikan merupakan penjumlahan sinyal kontrol dari kontroler dan bisa sebesar *massa* dikali gaya gravitasi, ditampilkan pada Gambar 3.23.



Gambar 3.23 Diagram Blok Z

Secara langsung, pengendalian sudut *roll*, *pitch* dan *yaw* dapat dilakukan dengan mengendalikan torsi U2, U3. Dari diagram blok yang disajikan pada Gambar 3.24, sudut adalah *roll, pitch,* dan *yaw* berturut-turut n=2,3,4.

Gambar 3.24 Diagram Blok Sudut

Proses mencari parameter kontroler untuk gerak rotasi dilakukan secara satu per satu menggunakan kontroler PID manual menggunakan cara linierisasi. Pada bagian gerak rotasi sudut roll dan pitch terjadi hubungan yang berkaitan, sehingga dilakukan prosedur:

1. Semua sudut dibuat 0 rad dan dicari parameter kontroler
2. Karena *quadcopter* merupakan *plant* non-linier dan kontroler PID adalah suatu sistem kontrol untuk plant linier, maka *plant quadcopter* harus di linearisasikan terlebih dahulu.
3. Saat mencari sudut *roll*, sudut yang lain dibuat 0 rad dan sebaliknya.

Dari model matematika *quadcopter* yang telah diperoleh, kemudian dibawa kedalam bentuk *state* dengan melakukan *linierisasi* pada *operating point*-nya. Maka didapatkan hubungan *linier* translasi adalah sebagai berikut :

$$\dot{x} = v_x \tag{3.87}$$

$$\dot{y} = v_y \tag{3.88}$$

$$\dot{z} = v_z \tag{3.89}$$

$$\dot{v_x} = (sin\psi sin\phi + cos\psi sin\theta cos\phi)\frac{U1}{m} \tag{3.90}$$

$$\dot{v_y} = (-cos\psi sin\phi + sin\psi sin\theta cos\phi)\frac{U1}{m} \tag{3.91}$$

$$\dot{v}_z = -g + (cos\theta cos\phi)\frac{U1}{m} \tag{3.92}$$

Sedangkan hubungan *linier* rotasi adalah

$$\dot{\phi} = p \tag{3.93}$$

$$\dot{\theta} = q \tag{3.94}$$

$$\dot{\psi} = r \tag{3.95}$$

$$\dot{p} = \frac{Iyy - Izz}{Ixx} qr + \frac{Jr}{Ixx} q\Omega + \frac{U2}{Ixx} \tag{3.96}$$

$$\dot{q} = \frac{Izz - Ixx}{Iyy} pr + \frac{Jr}{Iyy} p\Omega + \frac{U3}{Iyy} \tag{3.97}$$

$$\dot{r} = \frac{Ixx - Iyy}{Izz} pq + \frac{U4}{Izz} \tag{3.98}$$

Sehingga bentuk *state* dari model *quadcopter* adalah

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{Vx} \\ \dot{Vy} \\ \dot{Vz} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & & & & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & & & \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ Vx \\ Vy \\ Vz \\ \phi \\ \theta \\ \psi \\ p \\ q \\ r \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & \\ & & & & 0 & \\ & & & & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \Omega \\ g \end{bmatrix} \tag{3.99}$$

Pada Persamaan 3.28, matrik A dan B yang masih kosong, dihitung dengan melakukan linierisasi pada *operating point* saat *quadcopter* terbang *hover*, untuk linierisasi gerak translasi berdasarkan Persamaan 3.19-3.21 adalah

$$\begin{bmatrix} \dot{Vx} \\ \dot{Vy} \\ \dot{Vz} \end{bmatrix} = f\,(\theta, \phi, \psi, u_1) \tag{3.100}$$

$$\begin{bmatrix} \dot{Vx} \\ \dot{Vy} \\ \dot{Vz} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} + \begin{bmatrix} b_{11} & b_{16} \\ b_{21} & b_{26} \\ b_{31} & b_{36} \end{bmatrix} \begin{matrix} u_1 \\ g \end{matrix} \tag{3.101}$$

Operating point quadcopter saat terbang *hover* untuk $\theta = 0,\ \phi = 0,\ \psi = 0$ sedangkan untuk $\frac{U_1}{m}$ adalah

$$U_1 = b(\Omega_1{}^2 + \Omega_2{}^2 + \Omega_3{}^2 + \Omega_4{}^2) \tag{3.102}$$

Maka :

$$a_{11} = \frac{\partial}{\partial \phi}(sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U1}{m}$$
$$= (sin\psi\ cos\phi - cos\psi\ sin\theta\ sin\phi)\frac{U1}{m} \quad (3.103)$$
$$a_{12} = \frac{\partial}{\partial \theta}(sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U1}{m} \quad (3.104)$$
$$a_{13} = \frac{\partial}{\partial \psi}(sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U1}{m} \quad (3.105)$$
$$a_{21} = \frac{\partial}{\partial \phi}(-cos\psi\ sin\phi + sin\psi\ sin\theta\ cos\phi)\frac{U1}{m}$$
$$= (-cos\psi\ cos\phi - sin\psi\ sin\theta\ sin\phi)\ \frac{U1}{m} \quad (3.106)$$
$$a_{22} = \frac{\partial}{\partial \theta}(-cos\psi\ sin\phi + sin\psi\ sin\theta\ cos\phi)\frac{U1}{m}$$
$$= (-cos\psi\ sin\phi + sin\psi\ cos\theta\ cos\phi)\frac{U1}{m} \quad (3.107)$$
$$a_{23} = \frac{\partial}{\partial \psi}(-cos\psi\ sin\phi + sin\psi\ sin\theta\ cos\phi)\frac{U1}{m}$$
$$= (sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U1}{m} \quad (3.108)$$
$$a_{31} = \frac{\partial}{\partial \phi}(-g + \cos\theta\ cos\phi)\frac{U1}{m} = (-cos\theta\ sin\phi)\frac{U1}{m} \quad (3.109)$$
$$a_{32} = \frac{\partial}{\partial \theta}(-g + \cos\theta\ cos\phi)\frac{U1}{m} = (-sin\theta\ cos\phi)\frac{U1}{m} \quad (3.110)$$
$$a_{33} = \frac{\partial}{\partial \psi}(-g + \cos\theta\ cos\phi)\frac{U1}{m} \quad (3.111)$$
$$b_{11} = \frac{\partial \dot{V}x}{\partial U_1} = \frac{\partial(sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U_1}{m}}{\partial U_1} \quad (3.112)$$
$$b_{21} = \frac{\partial \dot{V}y}{\partial U_1} = \frac{\partial(-cos\psi\ sin\phi + sin\psi\ sin\theta\ cos\phi)\frac{U_1}{m}}{\partial U_1} \quad (3.113)$$
$$b_{31} = \frac{\partial \dot{V}z}{\partial U_1} = \frac{\partial(-g + \cos\theta\ cos\phi)\frac{U_1}{m}}{\partial U_1} = (0 + \cos\theta\ cos\phi)\frac{1}{m} \quad (3.114)$$
$$b_{16} = \frac{\partial \dot{V}x}{\partial g} = \frac{\partial(sin\psi\ sin\phi + cos\psi\ sin\theta\ cos\phi)\frac{U_1}{m}}{\partial g} \quad (3.115)$$
$$b_{26} = \frac{\partial \dot{V}y}{\partial g} = \frac{\partial(-cos\psi\ sin\phi + sin\psi\ sin\theta\ cos\phi)\frac{U_1}{m}}{\partial g} \quad (3.116)$$
$$b_{36} = \frac{\partial \dot{V}z}{\partial g} = \frac{\partial(-g + \cos\theta\ cos\phi)\frac{U_1}{m}}{\partial g} \quad (3.117)$$

Sedangkan untuk *linierisasi* gerak rotasi berdasarkan Persamaan 3.25-3.27 adalah

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = f\,(p, q, r, \Omega, u_2, u_3, u_4) \tag{3.118}$$

Operating point quadcopter saat terbang *hover* untuk $p = 0$, $q = 0$, $r = 0$ dan $\Omega = 0$, dengan *operating point* tersebut saat dimasukkan kedalam Persamaan 3.96-3.98, maka matrik A akan bernilai nol semua dan matrik B akan bernilai seperti pada Persamaan 3.119

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} + \begin{bmatrix}\frac{1}{ixx} & 0 & 0 & 0\\ 0 & \frac{1}{iyy} & 0 & 0\\ 0 & 0 & \frac{1}{iyy} & 0\end{bmatrix}\begin{bmatrix}u_2\\ u_3\\ u_4\\ \Omega\end{bmatrix} \tag{3.119}$$

Dengan menggabungkan *state* pada Persamaan 3.99 dengan Persamaan 3.101, 3.118 dan 3.119 sehingga diperoleh *state* dari model *quadcopter* adalah

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{Vx}\\ \dot{Vy}\\ \dot{Vz}\\ \dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\\ \dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{u1}{m} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{u1}{m} & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ Vx\\ Vy\\ Vz\\ \phi\\ \theta\\ \psi\\ p\\ q\\ r\end{bmatrix} + \begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ \frac{1}{m} & 0 & 0 & 0 & 0 & -1\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & \frac{1}{ixx} & 0 & 0 & 0 & 0\\ 0 & 0 & \frac{1}{iyy} & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1}{izz} & 0 & 0\end{bmatrix}\begin{bmatrix}u_1\\ u_2\\ u_3\\ u_4\\ \Omega\\ g\end{bmatrix} \tag{3.120}$$

Perhitungan mencari nilai matriks dari persamaan diatas menggunakan persamaan matriks jacobian. Metode iterasi jacobian merupakan salah satu bidang analisis numerik yang digunakan untuk menyelesaikan permasalahan persamaan linear. Dari hasil perhitungan linierisasi dari persamaan diatas, maka akan di dapatkan nilai parameter kontrol PID untuk gerak rotasi sudut *roll* dan sudut *pitch*. Hasil nilai parameter ditampilkan pada Tabel 3.2.

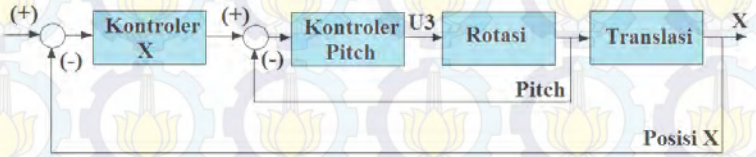
Tabel 3.2 Parameter PID Gerak Rotasi *Pitch* dan *Roll*

Gain	Gerak Rotasi	
	Pitch	***Roll***
Kp	10	10
Ki	0,001	0,001
Kd	3	3

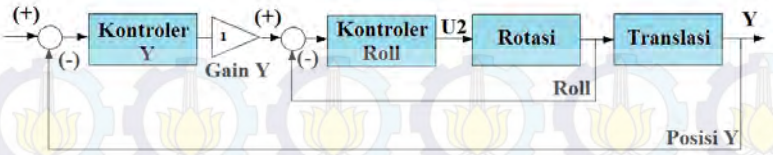
3.8.2 Perancangan Kontroler PID-*Fuzzy* Untuk Gerak Translasi Sumbu X dan Sumbu Y

Untuk mendapatkan hasil respon yang optimal maka dilakukan *tuning* parameter kontrol sehingga diperoleh respons *output* seperti yang diharapkan pada saat menentukan kriteria desain. Dalam melakukan perancangan kontroler, harus diketahui terlebih dahulu sinyal yang dapat dimanipulasi untuk bisa mendapatkan respon yang diinginkan. Respon yang diinginkan adalah *quadcopter* mampu bergerak menuju titik koordinat pada sumbu X dan sumbu Y dengan menjaga kestabilan gerak *pitch* maupun *roll*.

Dengan mengkategorikan persamaan percepatan X, Y, dan Z adalah persamaan translasi, serta persamaan Ø, Θ, dan ψ adalah persamaan rotasi. Dari rotasi *pitch* dan *roll*, gerak translasional pada sumbu X dan sumbu Y *quadrotor* dipengaruhi oleh gaya *thrust* dan dapat dikendalikan dengan mengatur sudut *pitch* dan -*roll* dari *quadrotor*. Maka akan diperoleh diagram blok *cascade* seperti pada Gambar 3.25 dan 3.26.



Gambar 3.25 Diagram Blok *Cascade* X



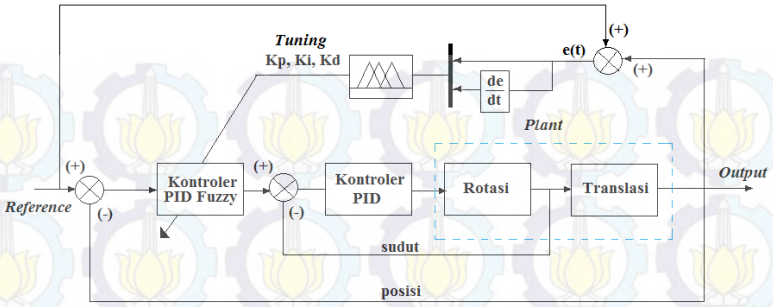
Gambar 3.26 Diagram Blok *Cascade* Y

Secara keseluruhan perancangan simulasi kontroler PID dan PID-Fuzzy pada *quadcopter* dapat ditunjukkan pada Gambar 3.27. Perancangan simulasi dilakukan pada *simulink* menggunakan perangkat lunak Matlab. Pada sistem navigasi *waypoint,* gerak *pitch* dan *roll* dijaga kestabilannya menggunakan kontroler PID, sedangkan untuk sumbu X dan sumbu Y pengaturannya menggunakan kontroler PID *fuzzy*.

Metode *tuning* disini digunakan untuk menentukan parameter Kp, Ki, dan Kd. Karena parameter *plant* yang selalu berubah-ubah sesuai dengan keadaan, maka kontroler PID-*Fuzzy* disini diharapkan mampu menjaga kestabilan gerak *quadcopter* saat melakukan gerak lateral, menyusuri lintasan yang telah direncanakan sebelumnya secara *online*. Dalam merancang kontroler PID-*Fuzzy* menggunakan *toolbox Fuzzy Logic Designer* yang telah tersedia dalam *software* Matlab 2014b.

1. Struktur Kontroler *Self-Tuning* PID *Fuzzy*

   Self tuning kontroler PID Fuzzy berarti bahwa tiga parameter PID yaitu Kp, Ki, dan Kd diperoleh dari hasil tunning Fuzzy. Koefisien dari konvensional PID tidak diatur dengan benar untuk plant non-linier dengan variasi parameter yang tidak terduga. Oleh karena itu, diperlukan *auto tunning* untuk menyesuaikan parameter PID. Struktur dari *self tuning* kontroler PID *Fuzzy* ditunjukkan pada Gambar 3.27



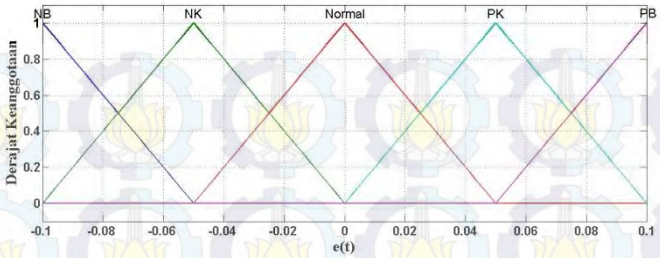
Gambar 3.27 Struktur Kontroler *Self Tuning* PID *Fuzzy*

Dimana e(t) adalah kesalahan antara yang dinginkan titik posisi *set* dan *output*, sedangkan de (t) adalah derivasi dari kesalahan. Parameter PID diatur dengan menggunakan *Fuzzy Inference* yang menyediakan pemetaan non-linier dari kesalahan dan derivasi dari kesalahan parameter.

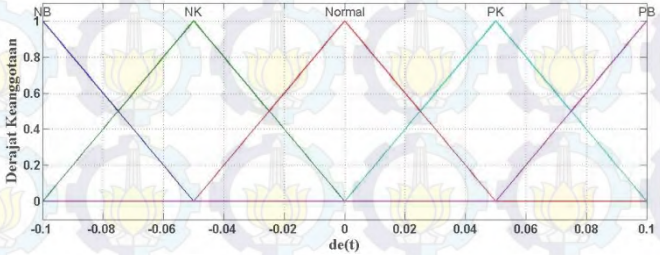
2. Desain Kontroler *Self Tuning* PID-Fuzzy

Aturan dirancang berdasarkan pada karakteristik dari aktuator *plant* dan sifat dari kontroler PID. Oleh karena itu, perancangan penalaran *input/output fuzzy* diperoleh dengan operasi agregasi *fuzzy*. Agregasi dan metode defuzzifikasi yang digunakan adalah penentuan *range* maksimal dan minimal sesuai rentang variabel parameter.

Mengenai struktur fuzzy, ada dua masukan untuk inferensi *fuzzy*, yaitu e(t) dan turunan dari kesalahan de(t), dan tiga output untuk setiap parameter kontroler PID, masing-masing Kp, Ki, dan Kd. Model Mamdani diterapkan sebagai struktur inferensi *Fuzzy* dengan beberapa modifikasi agar mendapatkan nilai terbaik untuk nilai parameter Kp, Ki, Kd. Masukan struktur inferenzy fuzzy e(t) dan de(t) ditampilkan pada Gambar 3.28 dan Gambar 3.29.



Gambar 3.28. Fungsi Keanggotaan e(t) Lima Anggota Himpunan



Gambar 3.29. Fungsi Keanggotaan de(t) lima anggota himpunan;

Dikarenakan tingkat keberhasilan dan kegagalan aksi kontrol sebuah *plant* tergantung dari desain sistem yang digunakan, maka desain kontroler harus sesuai dengan karakteristik dan spesifikasi *plant*. Sehingga di harapkan, saat kontroler melakukan aksi kontrol nilai error yang dihasilkan tidak lebih dari 0.1%. Kisaran masukan ini dari -0,1 sampai 0,1 yang diperoleh dari nilai absolut dari kesalahan sistem dan turunannya melalui *gain*. Kisaran ini merupakan batas error plant yang di desain dari sistematika metode *fuzzy*, sehingga nantinya pada saat sistem berjalan, kontroler mampu menjaga kestabilan gerak *quadcopter* saat melakukan gerak lateral *way-to-way point* menyusuri lintasan yang telah di tentukan.

Fungsi keanggotaan yang digunakan pada masukan kontroler adalah fungsi keanggotaan segitiga. Variabel masukan e(t) dan de(t) yang digunakan adalah lima anggota himpunan pendukung *fuzzy* seperti yang ditunjukkan oleh Gambar 3.28, yaitu NB (negatif besar), NK (negatif kecil), N (normal), PK (positif kecil), dan PB (positif besar). Dasar aturan *rule base* lima anggota himpunan pendukung *fuzzy* yang

digunakan pada kontrol gerak translasi *way-to-way point* ditampilkan pada Tabel 3.3

Tabel 3.3 Aturan *Rule Base* Lima Anggota Himpunan Fuzzy Pendukung

de(t) / e(t)	NB	NK	N	PK	PB
NB	NB	NB	NK	NK	N
NK	NB	NK	NK	N	PK
N	NK	NK	N	PK	PK
PK	NK	N	PK	PK	PB
PB	N	PK	PK	PB	PB

Umumnya, pengaturan aturan *fuzzy* tergantung pada *plant* dan jenis kontrolernya. Berkaitan dengan pengaturan variabel *input/output, rule base* aturan fuzzy ditampilkan pada Tabel 3.3 dan penyusunan *rule base fuzzy* ditampilkan pada Gambar 3.33 terdiri sebagai berikut :

" Aturan I : Jika e(t) adalah A1 dan de(t) adalah A2 maka K´p = Bi dan K´I = Cid dan K´d = Di. Dimana i = 1,2,3,…, n, dan n adalah jumlah aturan."

Dalam pembuatan desain kontroler tugas akhir ini telah didapatkan 25 *rule base fuzzy* untuk lima anggota himpunan. Dalam mendesain rentang nilai parameter kontroler, rentang variabel parameter Kp, Ki, dan Kd masing-masing adalah [Kp min ; Kp max], [Ki min ; Ki max], dan [Kd min ; Kd max]. Kisaran setiap parameter ditentukan berdasarkan simulasi kontroler PID untuk memperoleh kemungkinan *rule base* dengan efisiensi inferensi yang tinggi. Kisaran setiap parameter adalah Kp ϵ [0,008 ; 0,012], Ki ϵ [0], dan Kd ϵ [0,058 ; 0,062].

Rentang nilai minimum dan maksimum diperoleh dari hasil pengujian sehingga di dapatkan hasil yang optimal. Rentang nilai Kp dan Kd diatur antara 0 sampai 1 sehingga batas maksimal untuk nilai penguatan kontroler PID adalah 1. Dikarenakan pada penelitian tugas akhir ini, sistem *plant* yang diinginkan adalah sistem *online atau real time* (perubahan nilai parameter kontroler berubah mengikuti perubahan *plant* saat melakukan *tracking waypoint*) maka dibuat sebuah rentang nilai yang dapat menjaga ke-optimalan aksi kontrol *quadcopter* saat

berjalan menyusuri lintasan. Desain rentang nilai yang digunakan pada kontroler merupakan nilai yang dianggap baik saat *quadcopter* melakukan aksinya.

Batas nilai tersebut di tentukan dari desain spesifikasi fuzzy yang dibuat, karena keberhasilan suatu *self tuning fuzzy* terhadap kontrol parameter Kp, Ki, Kd ditentukan oleh spesifikasi model yang kita desain. Parameter dapat dikalibrasi selama rentang *interval* yang diberikan sebagai berikut :

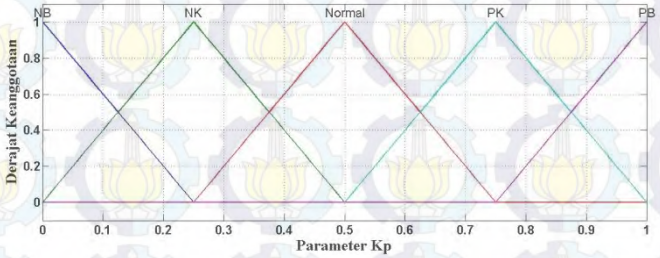
$$K'p = \frac{Kp - Ki\,min}{Kp\,max - Kp\,min} = \frac{Kp - 0{,}008}{0{,}012 - 0{,}008} \tag{3.121}$$

$$K'i = \frac{Ki - Ki\,min}{Ki\,max - Ki\,min} = 0 \tag{3.122}$$

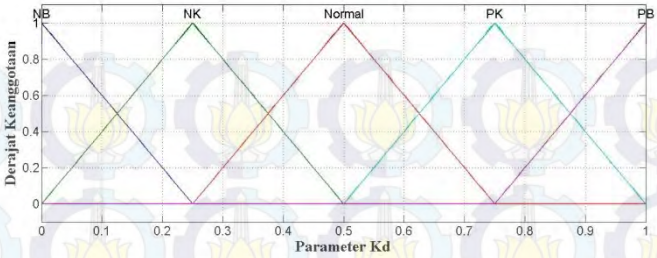
$$K'd = \frac{Kd - Kd\,min}{Kd\,max - Kd\,min} = \frac{Kd - 0{,}058}{0{,}062 - 0{,}058} \tag{3.123}$$

Sehingga kita dapatkan Kp = 0,004 K´p + 0,008, Ki = 0, dan Kd = 0,004 K´d + 0,058.

Fungsi keanggotaan himpunan pendukung dalam menentukan nilai parameter Kp dan Kd saat di *tuning* menggunakan *fuzzy* ditampilkan pada Gambar 3.30 dan Gambar 3.31.



Gambar 3.30. Fungsi Keanggotaan Kp Lima Anggota Himpunan



Gambar 3.31. Fungsi Keanggotaan Kd Lima Anggota Himpunan

Fungsi keanggotaan yang digunakan pada masukan kontroler Kp dan Kd adalah fungsi keanggotaan segitiga. Variabel keluaran Kp dan Kd yang digunakan adalah lima anggota himpunan pendukung *fuzzy*. Fungsi keanggotaan untuk nilai parameter kontroler Kp dan Kd memiliki karakteristik variabel pemetaan yang sama yaitu NB (negatif besar), NK (negatif kecil), N (normal), PK (positif kecil), dan PB (positif besar). Dari hasil pengujian *tracking waypoint*, didapatkan hasil tuning parameter kontrol Kp dan Kd di tampilkan pada Tabel 3.5.

Tabel 3.4 Hasil *Tuning Fuzzy* Parameter Kontrol Kp dan Kd Gerak Translasi *Tracking Waypoint* Tanpa *Noise*

Sumbu Koordinat (X,Y)	Sumbu X		Sumbu Y	
	Kp	Kd	Kp	Kd
Titik (0,0)	0,02	0	0.02	0
Titik (2,2)	-0,04	-0,00012	$2{,}2 \times 10^{-5}$	-0,00019
Titik (-2,2)	-0,00026	0,00103	-0,039	0,0026
Titik (-2,-2)	0,04	-0,00027	-0,00018	0,00083
Titik (2,-2)	0,00018	-0,00087	0,04	0,00075
Titik (2,2)	$-2{,}5 \times 10^{-5}$	$4{,}5 \times 10^{-5}$	0,00054	-0,0018

Dari hasil perancangan kontroler PID Fuzzy, diperoleh penalaan parameter PID bahwa *tuning fuzzy* berhasil menjaga kestabilan *quadcopter* saat melakukan gerak lateral menempuh titik koordinat sumbu X dan sumbu Y yang telah di tentukan. Untuk menguji desain kontrol yang telah di rancang, maka diperlukan sebuah *noise* atau gangguan saat *quadcopter* berjalan menyusuri lintasan. Hal ini bertujuan

untuk mengetahui hasil respons dari *quadcopter* saat diberi gangguan apakah masih dapat mempertahankan posisi saat menyusuri lintasan atau sebaliknya, bergerak keluar jauh dari lintasan yang telah di buat. Dalam simulasi,untuk memberikan sinyal gangguan maka digunakan *random noise* yang akan memberikan sinyal gangguan terhadap sinyal kontrol secara acak, karena dalam kondisi nyata, terdapat gangguan eksternal seperti hembusan angin yang dapat mengganggu kinerja dari *quadcopter*.

Dari hasil pengujian desain kontrol terhadap sinyal gangguan yang diberikan, maka didapatkan nilai tuning parameter kontrol Kp dan Kd yang ditampilkan pada Tabel 3.6.

Tabel 3.5 Hasil *Tuning Fuzzy* Parameter Kp, Ki, dan Kd Gerak Translasi *Tracking Waypoint* Dengan *Noise* (Sinyal Gangguan)

Sumbu Koordinat (X,Y)	Sumbu X		Sumbu Y	
	Kp	Kd	Kp	Kd
Titik (0,0)	0,02	0	0,02	0
Titik (2,2)	-0,039	0,015	0,001	-0,0007
Titik (-2,2)	-0,00015	0,00083	-0,039	0,00085
Titik (-2,-2)	0,04	-0,0012	-0,00032	-0,0006
Titik (2,-2)	0,0015	-0,0012	0,037	-0,015
Titik (2,2)	0,00095	0,00084	-0,0008	-0,0015

Dari hasil perancangan kontroler PID *Fuzzy* dengan ditambah *noise* (sinyal gangguan), diperoleh penalaan parameter PID bahwa *tuning fuzzy* mampu menjaga *quadcopter* berjalan menyusuri lintasan meskipun sedikit keluar dari jalur lintasan yang telah dibuat.

3.9 Perancangan Sistem Perangkat Lunak

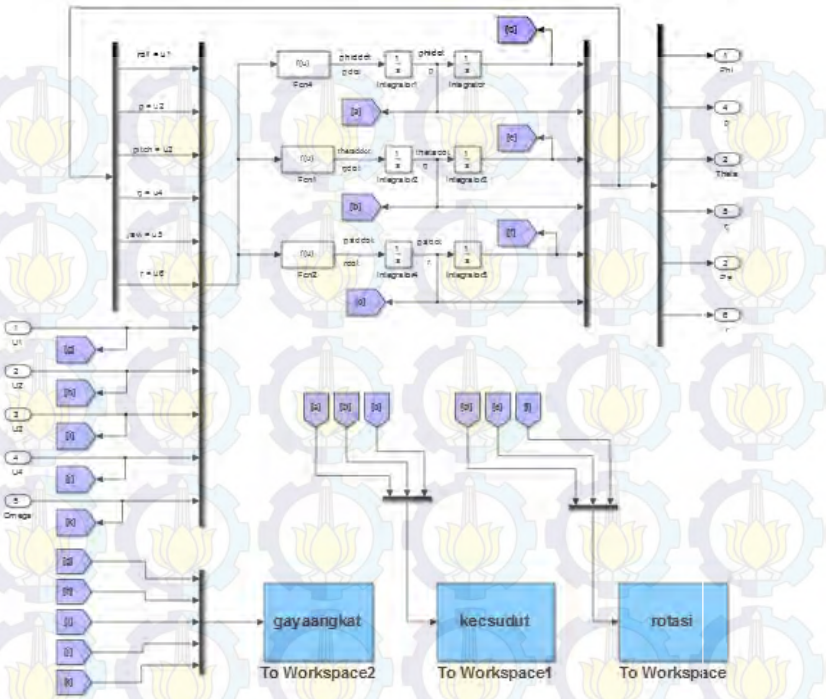
Dalam perancangan sistem perangkat lunak ini, diperlukan *software* yang digunakan sebagai penunjang *quadcopter* baik dalam hal perancangan simulasi maupun perencanaan implementasi. *Software* yang digunakan adalah Mission Planner dan Matlab 2014b.

3.9.1 *Software* Matlab 2014a

MATLAB (*matrix laboratory*) adalah perangkat komputasi numerikal dan bahasa pemrograman komputer yang memungkinkan manipulasi matriks, pem-plot-an fungsi dan data, implementasi algoritma, dan pembuatan antarmuka pengguna. Selain itu, dalam *software* Matlab terdapat sistem model Simulink yang dapat digunakan untuk mendesain suatu sistem. Dalam penelitian tugas akhir ini, *software* Matlab digunakan untuk men-simulasikan data identifikasi parameter dan model matematis *plant* sesuai dengan desain spesifikasi *quadcopter* yang telah di buat. Hal ini digunakan untuk memeriksa aksi kontrol terhadap *plant* apakah sudah sesuai dengan yang diharapkan.

3.9.1.1 Gerak Rotasi Sudut *Roll* dan *Pitch*

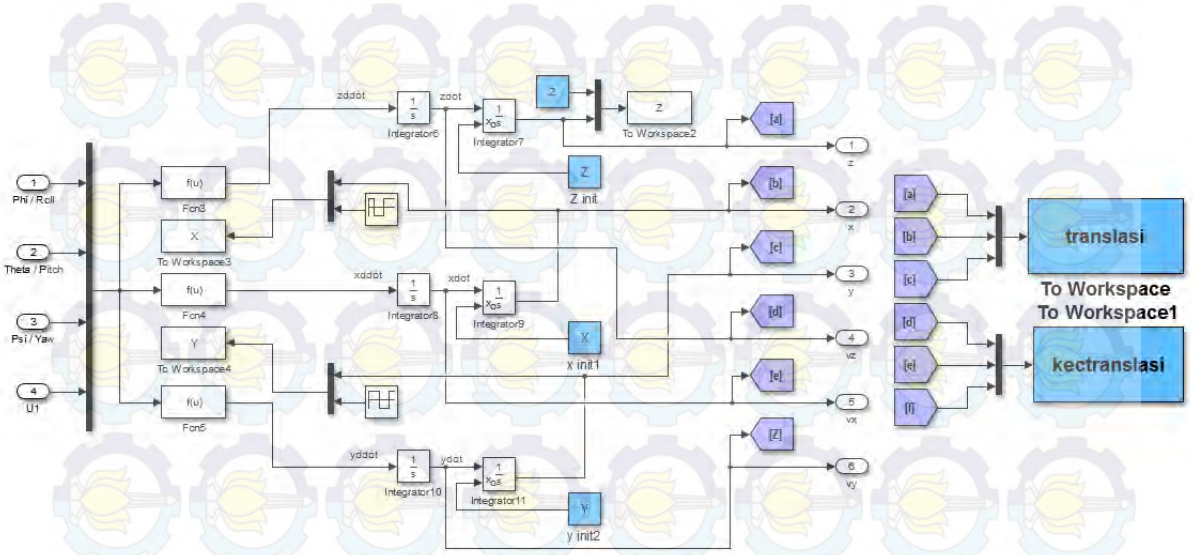
Gerak rotasi adalah gerak perputaran sumbu terhadap porosnya (gerak melingkar terhadap titik pusat *frame body quadcopter),* dimana dalam hal ini terdapat tiga sumbu yang menjadi arah gerakan yaitu sumbu X, sumbu Y, dan sumbu Z. Sumbu-sumbu ini merepresentasikan arah gerak sudut-sudut pada *quadcopter*. Dalam model matematis *quadcopter* terdapat tiga sudut rotasi yang digunakan yaitu sudut *roll, pitch, yaw*. Dalam desain simulasi Matlab, pengaturan gerakan sudut ini di buat dalam bentuk persamaan yang diubah dalam bentuk blok Simulink seperti di tampilkan pada Gambar 3.32.



Gambar 3.32 Diagram Blok Simulink Persamaan Gerak Rotasi

3.9.1.2 Gerak Translasi Sumbu X dan Sumbu Y

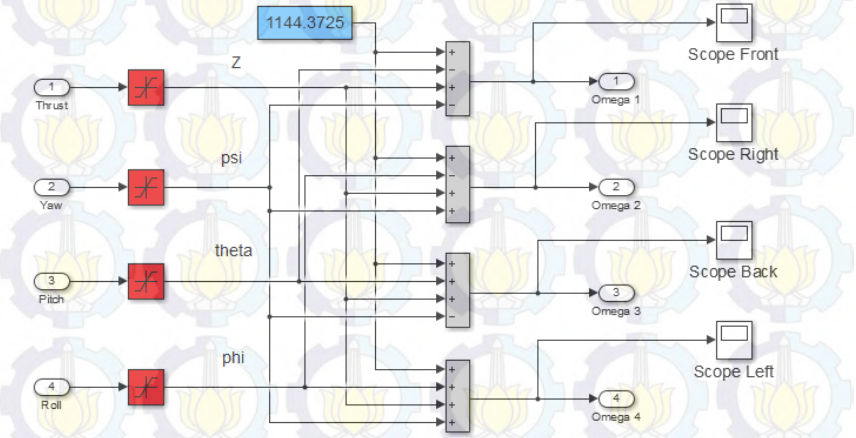
Gerak translasi merupakan gerak pergeseran suatu benda dengan bentuk dan lintasan yang sama. Untuk menentukan lintasan gerak *quadcopter*, maka digunakan sumbu koordinat X dan koordinat Y. Dalam tugas akhir ini lintasan *quadcopter* dibuat persegi (kotak) sehingga diharapkan *quadcopter* mampu bergerak menyusuri lintasan dengan stabil. Dalam desain simulasi Matlab, pengaturan gerakan translasi ini di buat dalam bentuk persamaan yang diubah dalam bentuk blok Simulink seperti di tampilkan pada Gambar 3.33.



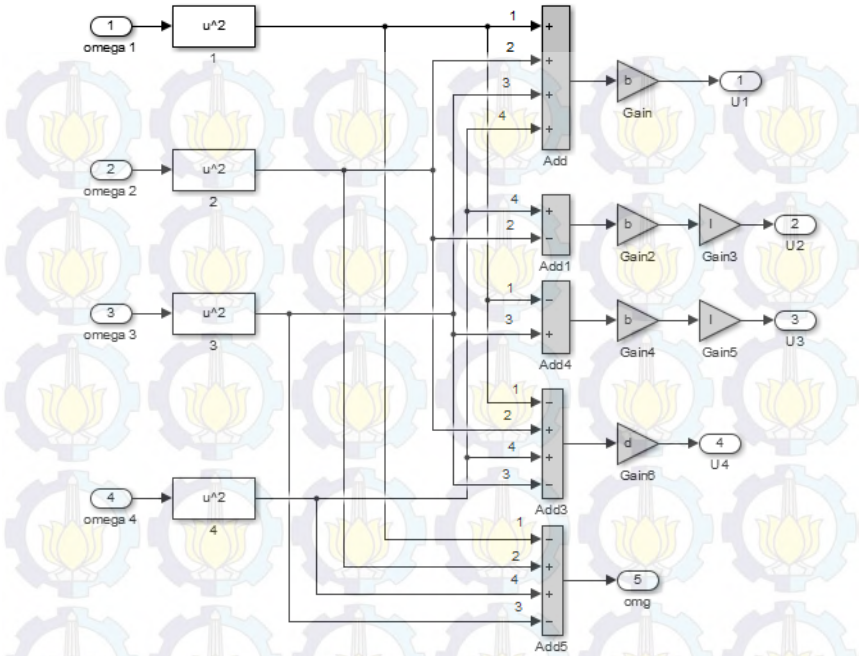
Gambar 3.33 Diagram Blok Simulink Persamaan Gerak Translasi

3.9.1.3 Persamaan Motor *Quadcopter*

Dalam desain simulasi Matlab, pengaturan gerakan motor ini di buat dalam bentuk persamaan yang diubah dalam bentuk blok Simulink seperti di tampilkan pada Gambar 3.34 dan Gambar 3.35.



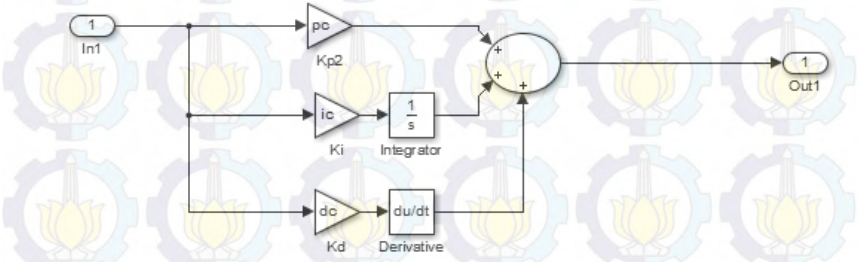
Gambar 3.34. Persamaan W to Omega



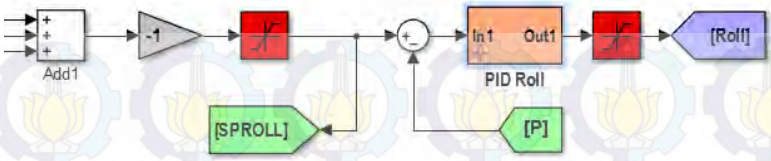
Gambar 3.35. Persmaan Omega to U

3.9.1.4 Desain Kontroler PID Untuk Gerak Rotasi

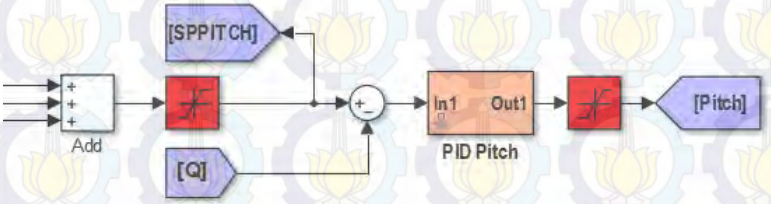
Desain kontrol ini dirancang untuk mengatur gerakan rotasi sudut *roll* dan *pitch* yang ditampilkan pada Gambar 3.36, Gambar 3.37, dan Gambar 3.38.



Gambar 3.36. Desain Rancangan Simulink Kontroler PID



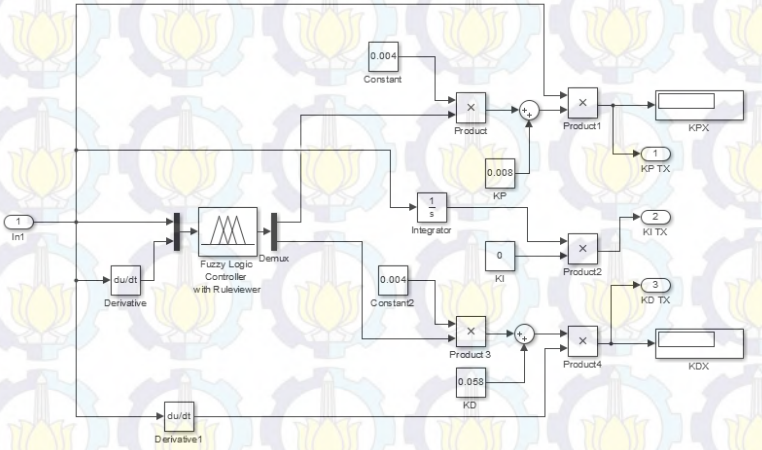
Gambar 3.37. Desain Model Simulink Sudut *Roll*



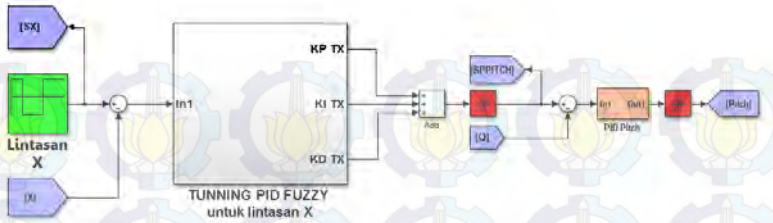
Gambar 3.38. Desain Model Simulink Sudut *Pitch*

3.9.1.5 Desain Rancangan Kontrol PID *Fuzzy* Untuk Gerak Translasi Sumbu X

Desain kontrol ini dirancang untuk mengatur gerakan translasi sumbu X yang ditampilkan pada Gambar 3.39 dan Gambar 3.40.



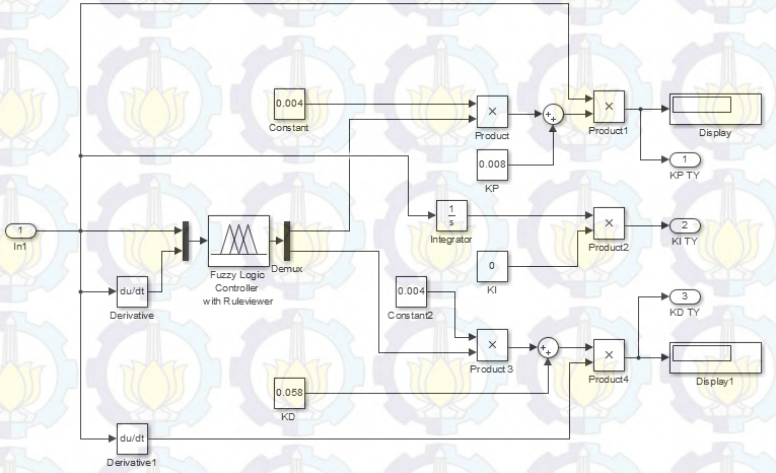
Gambar 3.39. Rancangan Kontroler PID *Fuzzy* Sumbu X



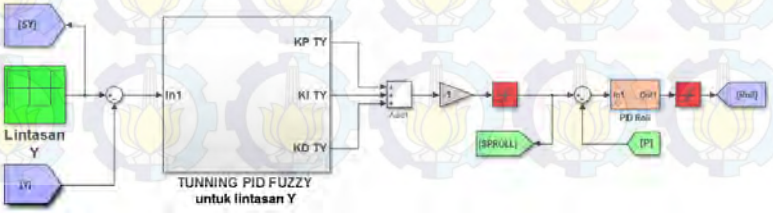
Gambar 3.40. Desain Simulink Kontroler PID *Fuzzy* Sumbu X

3.9.1.6 Desain Rancangan Kontrol PID *Fuzzy* Untuk Gerak Translasi Sumbu Y

Desain kontrol ini dirancang untuk mengatur gerakan translasi sumbu Y yang ditampilkan pada Gambar 3.41. dan Ga,bar 3.42.



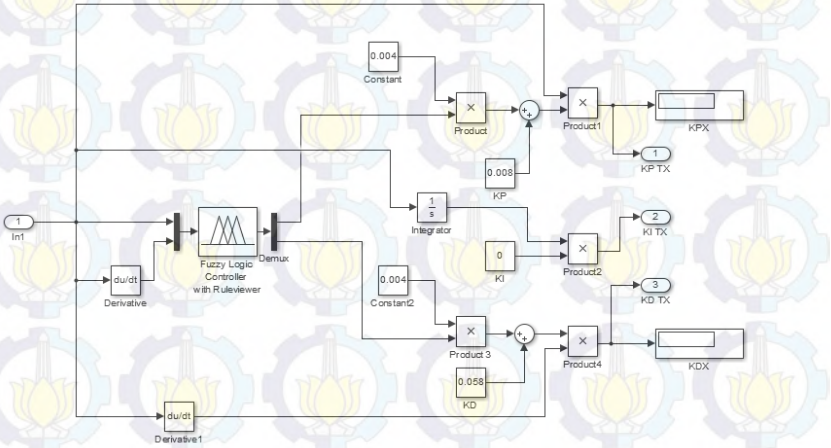
Gambar 3.41. Rancangan Kontroler PID *Fuzzy* Sumbu Y



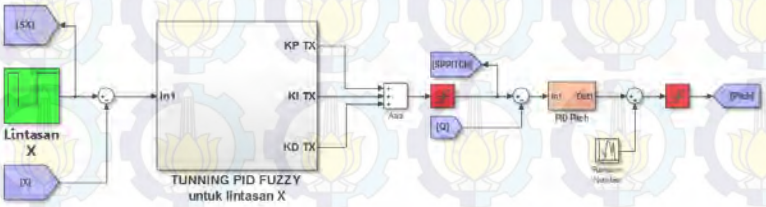
Gambar 3.42. Desain Simulink Kontroler PID *Fuzzy* Sumbu Y

3.9.1.7 Desain Rancangan Kontrol PID Fuzzy Untuk Gerak Translasi Sumbu X Dengan *Noise* (Sinyal Gangguan)

Desain kontrol ini dirancang untuk mengatur gerakan translasi sumbu X yang ditampilkan pada Gambar 3.43 dan Gambar 3.44.



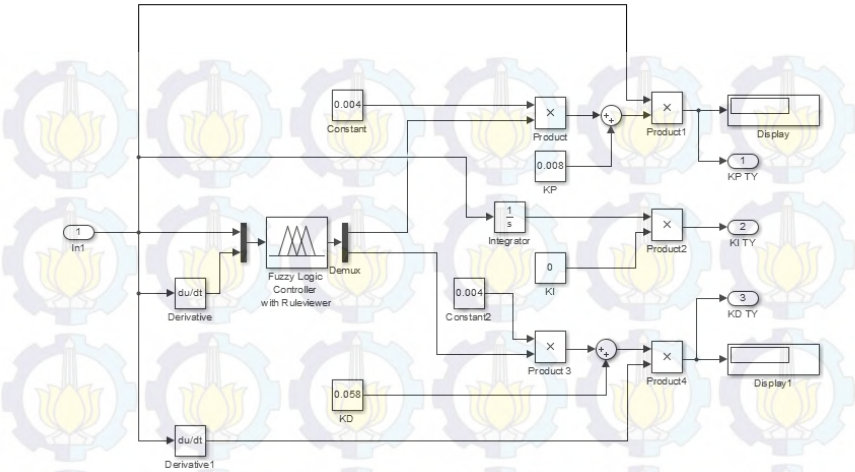
Gambar 3.43. Rancangan Kontroler PID *Fuzzy* Sumbu X Dengan *Noise*



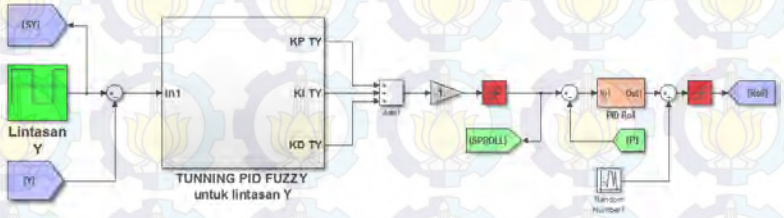
Gambar 3.44. Desain Simulink Kontroler PID Fuzzy Sumbu X Dengan *Noise*

3.9.1.8 Desain Rancangan Kontrol PID Fuzzy Untuk Gerak Translasi Sumbu Y Dengan *Noise* (Sinyal Gangguan)

Desain kontrol ini dirancang untuk mengatur gerakan translasi sumbu Y yang ditampilkan pada Gambar 3.45 dan Gambar 3.46.



Gambar 3.45. Rancangan Kontroler PID *Fuzzy* Sumbu X Dengan *Noise*



Gambar 3.46. Desain Simulink Kontroler PID Fuzzy Sumbu Y Dengan *Noise*

3.9.1.9 Desain Rancangan Simulasi Keseluruhan Sistem

Secara keseluruhan perancangan sistem gerak lateral *tracking waypoint* pada simulink matlab di tampilkan pada Lampiran B.4 dan Lampiran B.5.

3.9.2 *Software* Mission Planner Versi 1.3.31

Software ini berfungsi untuk kalibrasi, pengaturan konfigurasi, dan pengambilan data melalui *grand station*. Kalibrasi adalah kegiatan untuk menentukan kebenaran konvensional nilai penunjukkan alat ukur dan bahan ukur dengan cara membandingkan terhadap standar ukur.

Tujuan kalibrasi adalah untuk memeriksa apakah semua komponen *quadcopter* sudah sesuai dengan spesifikasi yang digunakan. Dalam hal ini, kalibrasi yang diperlukan adalah kalibrasi ESC (*Electronic Speed Controller*), kompas, dan *Remote Control*. Setelah dilakukan kalibrasi, quadcopter. Dalam penelitian tugas akhir ini, konfigurasi *quadcopter* yang digunakan adalah konfigurasi plus. Berikut ini adalah tampilan *software* mission planner pada Gambar 3.47.

Gambar 3.47 *Software* Mission Planner

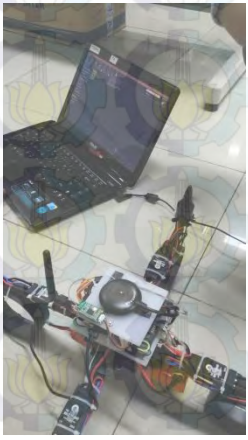
3.9.2.1 Pengaturan Konfigurasi *Quadcopter*

Sebelum melakukan kalibrasi terhadap *quadcopter*, terlebih dahulu kita melakukan pengaturan konfigurasi *quadcopter* yang sudah dirancang baik dalam hal konfigurasi plus, *setting remote control, flight controller. Compass,* dan lain-lain ditampilkan pada Gambar 3.48..

Gambar 3.48. *Configuration Setting*

3.9.2.2 Kalibrasi ESC Motor *Quadcopter*

Setelah mengatur konfigurasi *quadcopter*, maka tahap selanjutnya adalah melakukan kalibrasi *quadcopter*. Kalibrasi ini dilakukan agar kecepatan keempat motor sama atau tidak berbeda jauh sehingga saat dilakukan uji terbang kestabilan *quadcopter* terjaga. Proses kalibrasi motor di tampilkan pada Gambar 3.49.



Gambar 3.49. Kalibrasi Motor *Quadcopter*

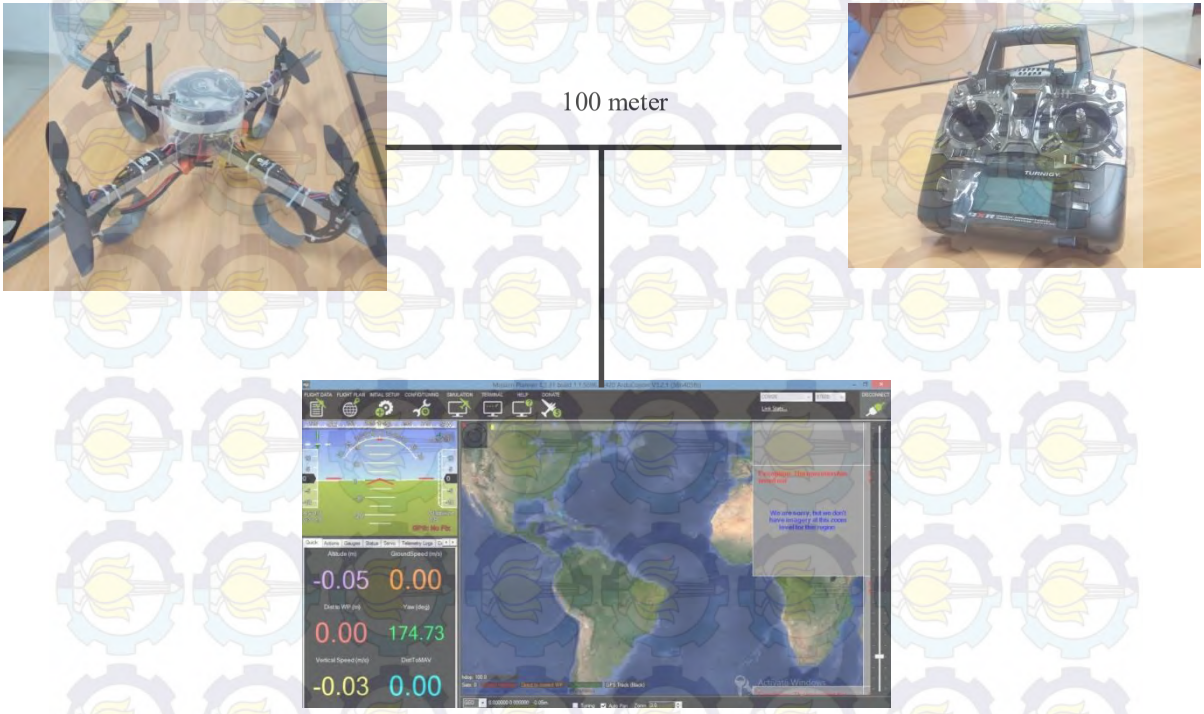
3.9.2.3 Kalibrasi Remote *Control*

Tahap selanjutnya adalah melakukan kalibrasi *remote control* yang digunakan sebagai pengendali manual *quadcopter*. Dalam hal ini diatur pergerakan *throttle*, *roll*, *pitch*, dan *yaw*. Proses kalibrasi *remote control quadcopter* ditampilkan pada Gambar 3.50.

Gambar 3.50. Kalibrasi *Remote Control*

3.9.2.4 Komunikasi Serial

Tahap terakhir adalah menguji komunikasi serial antara *quadcopter, remote control* dan *ground station*. Hal ini bertujuan untuk mengetahui seberapa jauh komunikasi yang apat dilakukan antara *quadcopter*, *remote control*, dan *ground station*. Dalam penggunaanya, komunikasi serial ini melakukan proses pengiriman dan penerimaan data. Pengiriman data dilakukan dari quadcopter saat terbang dan secara langsung diterima oleh *grand station,* sehingga dapat diketahui nilai dari sudut *roll*, *pitch*, *yaw*, *throttle*, dan lain-lain Uji coba komunikasi Serial di tampilkan pada Gambar 3.51.



Gambar 3.51. Komunikasi Serial *Quadcopter*

halaman ini sengaja dikosongkan

BAB IV
PENGUJIAN DAN ANALISIS SISTEM

Pengujian dan analisa sistem gerak lateral *way-to-way point* terdiri dari pengujian perangkat keras, analisa hasil bacaan sensor. perbandingan respons sistem dengan dan tanpa kontroler, hasil simulasi, dan implementasi yang dilakukan. Hasil pengujian dan simulasi sitem kemudian dianalisis untuk membandingkan hasilnya dengan hasil yang seharusnya.

4.1 Pengujian Motor BLDC *Quadcopter*

Salah satu cara untuk mengetahui karakteristik motor *brushless* dc adalah dengan metode modulasi lebar pulsa atau *pulse width modulation*. Dalam pengujiannya mikrokontroler yang digunakan adalah arduino yang memiliki alokasi data 8bit, atau memiliki rentang data (bit) dari 0-255 (2^8 = 256). Rentang data inilah yang nantinya di masukkan pada program arduino untuk melihat respons dari kecepatan putar motor. Semakin tinggi nilai *duty cycle* yang diberikan maka kecepatan motor akan bertambah, begitupun sebaliknya. Resolusi yang dimaksud yaitu rentang data (range) yang mampu di baca oleh mikrokontroler terhadap nilai PWM-nya. Hubungan kecepatan motor dengan nilai *range* data (bit) modulasi PWM akan di tampilkan pada Tabel 4.1.

Tabel 4.1 Hubungan Kecepatan Motor Dengan Modulasi PWM

Range Data (bit)	RPM Motor			
	Motor 1	Motor 2	Motor 3	Motor 4
75	1094	1069	1045	1086
80	1455	1458	1422	1472
85	1792	1843	1794	1850
90	2095	2152	2118	2192
95	2415	2466	2453	2520
100	2685	2780	2745	2824
105	2993	3112	3080	3155
110	3270	3397	3355	3442

Range Data (bit)	RPM Motor			
	Motor 1	Motor 2	Motor 3	Motor 4
115	3534	3678	3640	3735
120	3814	3979	3940	4041
125	4080	4271	4210	4320
130	4334	4507	4455	4542
135	4578	4781	4748	4859
140	4799	4815	4959	5094
145	5021	5004	5196	5334
150	5222	5244	5408	5568
155	5451	5467	5488	5793
160	5642	5688	5664	6026
165	5843	5877	5828	6251
170	6048	6014	5998	6268
175	6222	6243	6218	6322
180	6431	6402	6424	6438
185	6462	6422	6458	6464
190	6430	6425	6439	6457
195	6429	6440	6430	6450
200	6430	6442	6428	6455

Dari Tabel 4.1 diatas diketahui bahwa motor mulai berputar saat diberi range data bit 75, kecepatan motor berada diatas 1000 rpm. Sedangkan kecepatan maksimal motor berada diatas 6000 rpm dengan input data bit 200.

4.2 Pengujian Sensor

Dalam bagian ini, sensor yang akan diuji adalah sensor *accelerometer* dan *gyroscope* yang akan digunakan untuk pembacaan sudut *roll* dan *pitch.* Pengujian dilakukan se-akurat mungkin untuk melihat apakah terdapat kesalahan pada pembacaan sensor atau

kesalahan teknis lainnya. Sensor *accelerometer* dan *gyroscope* yang diuji sudah tertanam pada *flight controller* APM Planner 2.6.

Proses pengujian sudut dilakukan dengan mengukur perubahan sudut dengan busur, kemudian melihat sudut yang terbaca oleh *flight controller* dengan menggunakan bantuan perangkat lunak *Mission Planner*. Variabel yang akan diukur terlebih dahulu adalah sudut *roll*. Pengujian sudut *roll* akan berubah saat quadcopter melakukan gerak menyamping kanan atau kiri. Dengan mempercepat atau memperlambat *propeller* pada sisi kiri, dan secara bersamaan memperlambat atau mempercepat *propeller* pada sisi kanan dan sebaliknya, maka akan diperoleh rotasi dengan kecepatan sudut untuk gerakan menyamping ke kanan atau ke kiri. Sehingga nantinya pada saat dilakukan gerak rotasi, di dapatkan hasil respon sudut *roll* adalah nol atau mendekati nol derajat yang akan di tampilkan pada Gambar 4.4. Hasil pengujian sudut *roll* dapat dilihat di Tabel 4.2. Berdasarkan data dari Tabel 4.2, sensor memiliki kesalahan yang relatif kecil dan sensor masih berfungsi dengan baik.

Tabel 4.2 Pengujian Sudut *Roll*

No	Pembacaan Busur (^{o})	Pembacaan Sensor (^{0})
1	-60	-62,0
2	-45	-46,8
3	-30	-31,0
4	-15	-14,8
5	0	0,10
6	15	14,8
7	30	31,6
8	45	46,8
9	60	61,0

Varibel kedua yang akan digunakan untuk menguji modul *flight controller* adalah sudut *pitch*. Pembacaan sensor akan dibandingkan dengan pengukuran busur untuk mengetahui apabila terdapat kesalahan pada pembacaan sensor kemudian hasilnya di bandingkan dengan hasil konfigurasi sensor pada ardupilot yang ada pada *software* Mission

Planner. Pengujian ini dilakukan untuk mengetahui besarnya perubahan sudut saat melakukan gerak rotasi dan gerak translasi terhadap sumbu X dan sumbu Y, karena sudut *pitch* akan dipertahankan pada nilai nol saat melakukan gerak lateral sehingga posisi *quadcopter* dapat stabil saat bergerak menyusuri lintasan. Mekanisme yang sama seperti gerak sudut *roll*, namun tingkat kecepatan atau perlambatan dilakukan pada *propeller* disisi depan dan belakang. Nantinya, sudut *pitch* ini akan berpengaruh terhadap tingkat kemiringan *quadcopter* saat bergerak maju dan mundur. Pada saat *quadcopter* melakukan gerak rotasi sudut *pitch*,. didapatkan hasil respon sudut *pitch* adalah nol atau mendekati nol derajat yang akan di tampilkan pada Gambar 4.5. Hasil pengukuran dari sensor *pitch* dapat dilihat pada Tabel 4.3. Sensor untuk sudut *pitch* berfungsi dengan baik meskipun terjadi kesalahan dalam pengukuran (perbandingan kesalahan antara sensor dengan pengukuran sebenarnya relatif kecil).

Tabel 4.3 Pengujian Sudut *Pitch*

No	Pembacaan Busur (^{o})	Pembacaan Sensor (^{0})
1	-60	-60,5
2	-45	-46,2
3	-30	-30,5
4	-15	-15,6
5	0	-0,30
6	15	15,1
7	30	32,0
8	45	46,2
9	60	61,0

4.3 Pengujian Komunikasi Serial

Pengujian komunikasi serial dilakukan melalui *telemetry* untuk menunjukkan efisiensi informasi yang dikirimkan. Telemetri merupakan sebuah teknologi pengukuran yang dilakukan dari jarak jauh dan melaporkan informasi kepada perancang atau operator sistem. Efisiensi data diperlukan untuk menghasilkan analisis yang *valid*. Pengujian dilakukan dengan mengirimkan nilai sensor dari *flight*

controller ke komputer sebagai *ground station* untuk mengetahui besarnya nilai gerak rotasi dan kecepatan motor saat *quadcopter* melakukan aksi *tracking waypoint*. Dari pengujian kalibrasi sensor dari modul *flight controller* diperoleh kesalahan data pengiriman 0% dengan waktu pencuplikan 13 ms dan baudrate 57600.

Selain itu, pengujian ini dilakukan untuk mengetahui sistem pertukaran transformasi data atau membandingkan data antara modul ardupilot dengan *plant quadcopter* sehingga didapatkan kalibrasi yang baik pada *quadcopter*. Kalibrasi merupakan tahapan paling penting dalam penelitian tugas akhir ini karena pada tahap inilah akan terlihat apakah *quadcopter* dapat terbang dengan stabil atau kacau. Kalibrasi yang dilakukan adalah kalibrasi ESC motor, kalibrasi kompas, kalibrasi *remote control*, dan kalibrasi sensor pada *quadcopter* seperti yang telah dijelaskan pada bab sebelumnya. Untuk pengujian kalibrasi ESC motor, baudrate yang di gunakan adalah 115200 sesuai dengan karakteristik *flight controller* yang digunakan. Gambar 4.1 menampilkan komunikasi serial telah terhubung dan berjalan dengan baik.

Gambar 4.1 Pengujian Komunikasi Serial

Dalam tugas akhir ini jarak maksimal yang digunakan untuk uji coba *quadcopter* sejauh 100 meter. Hasil pengujian komunikasi serial *quadcopter* dengan *ground station* ditampilkan pada Tabel 4.4. Komunikasi serial masih dapat berjalan dengan baik saat melakukan pengiriman dan penerimaan data.

Tabel 4.3 Hasil Pengujian Komunikasi Serial

Jarak (meter)	Komunikasi
5	baik
10	baik
20	baik
30	baik
40	baik
50	baik
60	baik
70	baik
80	baik
90	baik
100	baik

4.4 Hasil Simulasi Gerak Rotasi dan Gerak Translasi

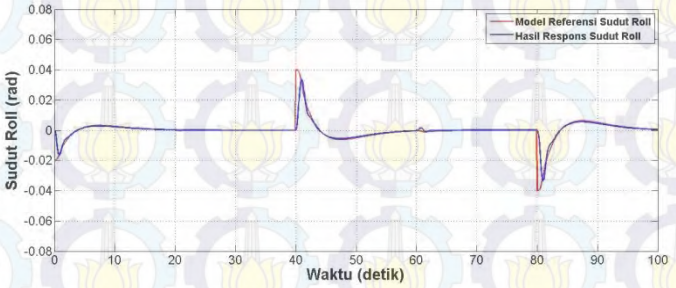
Simulasi sistem seluruhnya dirancang menggunakan perangkat lunak *Simulink Matlab.* Pada simulasi ini *quadcopter* mampu menjalankan sistem menuju titik koordinat, dan menjaga kestabilan saat melakukan gerak lateral *way-to-way point* menggunakan kontrol PID-*Fuzzy*. Parameter kontroler PID diperoleh dari hasil *auto-tunning Fuzzy* yang selalu berubah-ubah nilai parameternya sesuai dengan perubahan *plant* karena sistem *plant* berjalan secara *online (real time)*. Dalam tugas akhir ini dilakukan dua buah pengujian simulasi, yaitu pengujian respons tanpa *noise* (sinyal gangguan) dan pengujian menggunakan *noise* (sinyal gangguan).

4.4.1 Hasil Pengujian Tanpa *Noise* (Sinyal Gangguan) Gerak Rotasi Sudut *Roll* dan Sudut *Pitch*

Hasil respons sinyal kontrol gerak rotasi sudut roll dan sudut pitch menggunakan kontroler PID.

4.4.1.1 Sudut *Roll*

Sinyal kontrol sudut *roll* nantinya akan berpengaruh terhadap gerak translasi sumbu X, karena *quadcopter* akan bergerak vertikal pada titik koordinat sumbu X. Sudut *roll* akan membuat *quadcopter* melakukan gerak rotasi ke arah samping kanan dan kiri, karena pada sumbu X *quadcopter* akan bergerak menyamping, kiri dan kanan sesuai dengan lintasan. Dalam hal ini, sistem kontrol sudut *roll* masuk dalam kategori *inner loop* (*loop* dalam) sehingga memiliki kecepatan tiga kali lebih cepat daripada *outer loop* (*loop* luar). Dalam tugas akhir ini digunakan sistem diagram blok *cascade* seperti yang telah di bahas pada bab sebelumnya. Hasil respons sinyal kontrol sudut *roll* ditampilkan pada Gambar 4.2.



Gambar 4.2. Respons Gerak Rotasi Sudut *Roll*

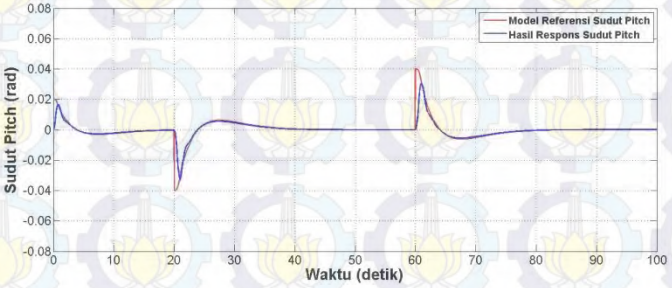
Dari hasil respons yang dihasilkan sinyal kontrol pada Gambar 4.2 terlihat bahwa kontroler mampu mendekati model referensi sudut roll yang diberikan. Meskipun respon yang dihasilkan sinyal kontrol gerak rotasi pada sudut *roll* masih terdapat kesalahan yang relatif kecil. Pada saat melakukan gerak rotasi sudut *roll* akan diikuti oleh perubahan nilai translasi sumbu X.

Dalam menjaga kestabilan gerak rotasi pada sudut *roll* dan *pitch* untuk melakukan gerak translasi, diketahui bahwa besar nilai yang diberikan gerak *rotasi* sudut *roll* selalu berbanding terbalik dengan besar nilai yang diberikan gerak *rotasi* sudut *pitch.*

4.4.1.2 Sudut *Pitch*

Sinyal kontrol sudut *pitch* nantinya akan berpengaruh terhadap gerak translasi sumbu Y, karena *quadcopter* akan bergerak horisontal

pada titik koordinat sumbu Y. Sudut *pitch* akan membuat *quadcopter* melakukan gerak rotasi ke arah depan dan belakang, karena pada sumbu Y *quadcopter* akan bergerak maju dan mundur sesuai dengan lintasan. Dalam hal ini, sistem kontrol sudut *pitch* masuk dalam kategori *inner loop* (*loop* dalam) sehingga memiliki kecepatan tiga kali lebih cepat daripada *outer loop* (*loop* luar). Dalam tugas akhir ini digunakan sistem diagram blok *cascade* seperti yang telah di bahas pada bab sebelumnya. Hasil respons sinyal kontrol sudut *pitch* ditampilkan pada Gambar 4.3.



Gambar 4.3. Respons Gerak Rotasi Sudut Pitch

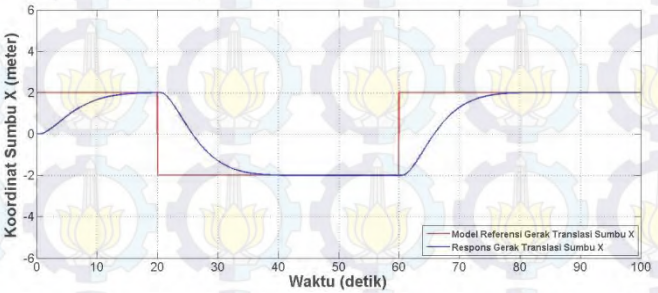
Dari hasil respons yang dihasilkan sinyal kontrol pada Gambar 4.3 terlihat bahwa kontroler mampu mendekati model referensi sudut *pitch* yang diberikan. Meskipun respon yang dihasilkan sinyal kontrol gerak rotasi pada sudut pitch masih terdapat kesalahan yang relatif kecil. Pada saat melakukan gerak rotasi sudut pitch akan diikuti oleh perubahan nilai translasi sumbu Y.

Dalam menjaga kestabilan gerak rotasi pada sudut *roll* dan *pitch* untuk melakukan gerak translasi, diketahui bahwa besar nilai yang diberikan gerak rotasi sudut *pitch* selalu berbanding terbalik dengan besar nilai yang diberikan gerak rotasi sudut *roll.*

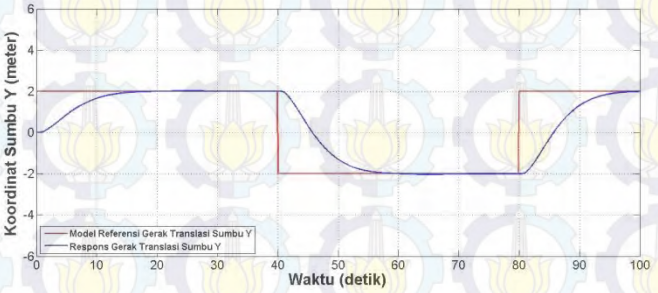
4.4.2 Hasil Pengujian Tanpa *Noise* (Sinyal Gangguan) Gerak Translasi Sumbu X dan Sumbu Y.

Perencanaan lintasan dibuat dari titik koordinat sumbu X dan Sumbu Y. Dalam tugas akhir ini, lintasan yang dibuat berbentuk kotak seperti pada Gambar 4.8. Q*uadcopter* akan berjalan vertikal dan horisontal menyusuri lintasan. Dalam hal ini, *quadcopter* akan bergerak maju, mundur, dan menyamping, kanan dan kiri.

Untuk menjaga kestabilan gerak translasi pada sumbu X dan sumbu Y, maka dalam tugas akhir ini digunakan kontroler PID Fuzzy. Metode *auto tuning fuzzy* digunakan untuk melakukan penalaan, menentukan nilai kontrol Kp dan Kd sistem saat *quadcopter* bergerak menyusuri lintasan dari titik awal sampai titik akhir. Dalam mendesain struktur *auto tuning fuzzy* ini, fungsi keanggotaan yang digunakan pada masukan kontroler adalah fungsi keanggotaan segitiga. Variabel masukan e(t) dan de(t) yang digunakan adalah lima anggota himpunan pendukung *fuzzy*. Hasil respons gerak translasi sumbu X dan sumbu Y ditampilkan pada Gambar 4.4 dan Gambar 4.5.



Gambar 4.4. Hasil Respons Gerak Translasi Sumbu X



Gambar 4.5. Hasil Respons Gerak Translasi Sumbu Y

Dari hasil respons diketahui bahwa sinyal kontroler gerak translasi mampu mendekati model referensi yang telah di tentukan. *Tuning Fuzzy* dalam menentukan nilai parameter kontroler terhadap *plant* yang selalu berubah-ubah kondisinya seiring dengan berjalannya *qadcopter* menyusuri lintasan menghasilkan respon yang baik. Hal ini terlihat dari hasil respons yang berhasil mendekati model referensi yang telah

ditentukan meskipun ada keterlambatan selama 2,2 detik terhadap *steady state*. Namun saat *quadcopter* berjalan menyusuri lintasan, hasil respon gerak translasi pada sumbu X dan Y masih terdapat kesalahan yang relatif kecil untuk menuju titik koordinat yang ditentukan yaitu sebesar ± 0,01 cm dan ± 0,035 cm.

Gerak translasi akan berpengaruh terhadap besarnya gerak rotasi sudut *roll* dan *pitch* pada *quadcopter*. Gerak rotasi terjadi lebih awal menuju sudut rotasi untuk menentukan arah gerak *quadcopter* kemudian disusul dengan gerak translasi untuk mencapai posisi *quadcopter* yang telah ditentukan.. Pada simulasi, penggunaan kontroler PID *Fuzzy* masih dinilai lambat untuk menghasilkan respon gerak lateral *way-to-way point* pada bidang sumbu X dan sumbu Y.

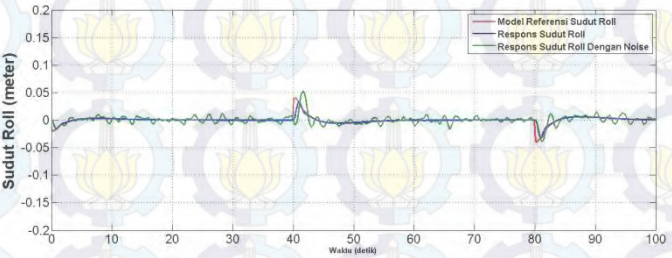
4.4.3 Hasil Pengujian Dengan *Noise* (Sinyal Gangguan) Gerak Rotasi Sudut *Pitch* dan Sudut *Roll*.

Untuk menguji desain kontrol PID yang telah dibuat, maka sistem akan diberi *noise* (sinyal gangguan) sehingga akan didapatkan hasil sinyal respons gerak rotasi sudut *pitch* dan sudut *roll*. Selain itu, pemberian *noise* (sinyal gangguan) bertujuan untuk mengetahui hasil respons dari *quadcopter* saat diberi gangguan apakah masih dapat mempertahankan posisi saat menyusuri lintasan atau sebaliknya, bergerak keluar jauh dari lintasan yang telah di buat. Dalam simulasi,untuk memberikan sinyal gangguan maka digunakan *random noise* yang akan memberikan sinyal gangguan terhadap sinyal kontrol secara acak, karena dalam kondisi nyata, terdapat gangguan eksternal seperti hembusan angin yang dapat mengganggu kinerja dari *quadcopter* saat terbang.

4.4.3.1 Sudut *Roll*

Sinyal kontrol sudut *roll* nantinya akan berpengaruh terhadap gerak translasi sumbu X, karena *quadcopter* akan bergerak vertikal pada titik koordinat sumbu X. Sudut *roll* akan membuat *quadcopter* melakukan gerak rotasi ke arah samping kanan dan kiri, karena pada sumbu X *quadcopter* akan bergerak menyamping, kiri dan kanan sesuai dengan lintasan. Sistem akan diberi *noise* (sinyal gangguan) untuk mengetahui apakah kontroler masih dapat menjaga kestabilan *quadcopter* saat terbang. Dalam hal ini, sistem kontrol sudut roll masuk dalam kategori *inner loop* (*loop* dalam) sehingga memiliki kecepatan tiga kali lebih cepat daripada *outer loop* (*loop* luar). Dalam tugas akhir ini digunakan

sistem diagram blok *cascade* seperti yang telah di bahas pada bab sebelumnya. Hasil respons sinyal kontrol sudut *roll* dengan noise (sinyal gangguan) ditampilkan pada Gambar 4.6.

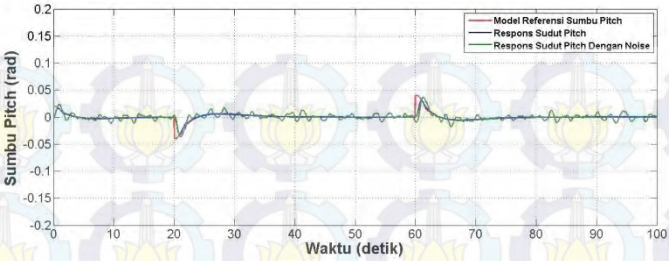


Gambar 4.6. Hasil Respons Gerak Rotasi Sudut Roll Dengan Noise

Dari hasil respons sinyal kontrol gerak rotasi pada Gambar 4.6, diketahui bahwa kontroler mampu menjaga kestabilan sudut *roll* saat diberi *noise* (sinyal gangguan). Sinyal kontrol mampu mendekati model referensi meskipun terdapat kesalahan, namun hasilnya masih dianggap baik.

4.4.3.2 Sudut *Pitch*

Sinyal kontrol sudut *pitch* nantinya akan berpengaruh terhadap gerak translasi sumbu Y, karena *quadcopter* akan bergerak horisontal pada titik koordinat sumbu Y. Sudut *pitch* akan membuat *quadcopter* melakukan gerak rotasi ke arah depan dan belakang, karena pada sumbu Y *quadcopter* akan bergerak maju dan mundur sesuai dengan lintasan. Sistem akan diberi *noise* (sinyal gangguan) untuk mengetahui apakah kontroler masih dapat menjaga kestabilan *quadcopter* saat terbang. Dalam hal ini, sistem kontrol sudut *pitch* masuk dalam kategori *inner loop* (*loop* dalam) sehingga memiliki kecepatan tiga kali lebih cepat daripada *outer loop* (*loop* luar). Dalam tugas akhir ini digunakan sistem diagram blok *cascade* seperti yang telah di bahas pada bab sebelumnya. Hasil respons sinyal kontrol sudut *pitch* ditampilkan pada Gambar 4.7.

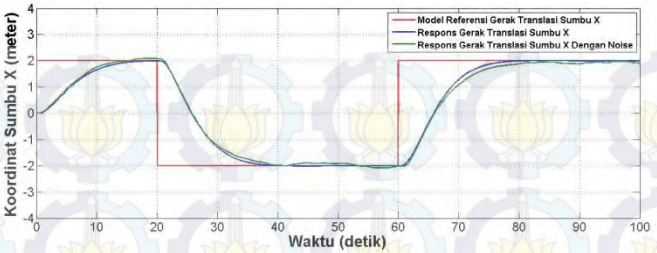


Gambar 4.7. Hasil Respons Gerak Rotasi Sudut *Pitch* Dengan *Noise*

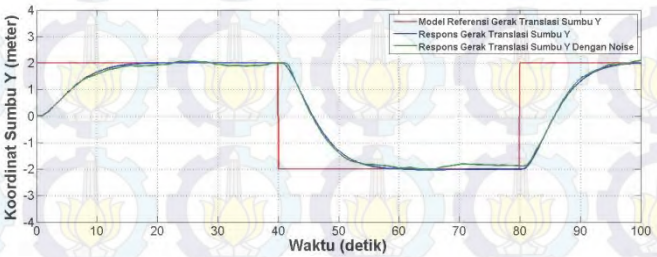
Dari hasil respons sinyal kontrol gerak rotasi pada Gambar 4.7, diketahui bahwa kontroler mampu menjaga kestabilan sudut *pitch* saat diberi *noise* (sinyal gangguan). Sinyal kontrol mampu mendekati model referensi meskipun terdapat kesalahan, namun hasilnya masih dianggap baik.

4.4.4 Hasil Pengujian Dengan *Noise* (Sinyal Gangguan) Gerak Translasi Sumbu X dan Sumbu Y.

Untuk menjaga kestabilan gerak translasi pada sumbu X dan sumbu Y, maka dalam tugas akhir ini digunakan kontroler PID Fuzzy. Sistem akan diberi *noise* (sinyal gangguan) untuk mengetahui apakah kontroler masih dapat menjaga kestabilan *quadcopter* saat terbang. Metode *auto tuning fuzzy* digunakan untuk melakukan penalaan, menentukan nilai kontrol Kp dan Kd sistem saat *quadcopter* bergerak menyusuri lintasan dari titik awal sampai titik akhir. Dalam mendesain struktur *auto tuning fuzzy* ini, fungsi keanggotaan yang digunakan pada masukan kontroler adalah fungsi keanggotaan segitiga. Variabel masukan e(t) dan de(t) dibagi menjadi lima dan tujuh himpunan pendukung *fuzzy*. Hasil respons gerak translasi sumbu X dan sumbu Y saat diberi *noise* (sinyal gangguan) ditampilkan pada Gambar 4.8. dan Gambar 4.9.



Gambar 4.8. Hasil Respons Gerak Translasi Sumbu X Dengan Noise



Gambar 4.9. Hasil Respons Gerak Translasi Sumbu Y Dengan Noise

Dari hasil respons sinyal kontrol gerak translasi sumbu X dan sumbu Y, diketahui bahwa kontroler mampu menjaga kestabilan *quadcopter* saat diberi *noise* (sinyal gangguan) ketika melakukan gerak lateral *tracking waypoint*. Sinyal kontrol mampu mendekati model referensi meskipun terdapat kesalahan yang relatif kecil, namun hasilnya masih dianggap baik.

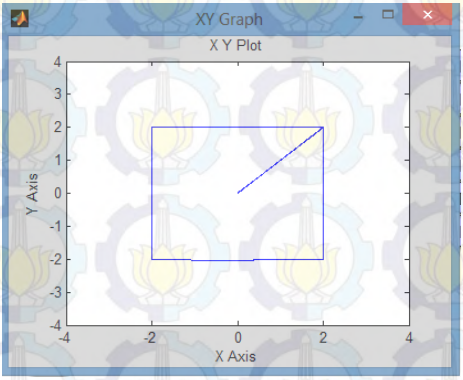
4.5 Hasil Simulasi 3D dan 2D

Pada simulasi sistem gerak lateral *way-to-way point*, terlebih dahulu ditentukan lintasan (titik koordinat) yang akan dilalui oleh *quadcopter*. Pada simulasi ini, untuk dapat melihat respon gerak translasi pada sumbu X dan Y maka dibuat lintasan berupa persegi (kotak) dengan ketentuan sebagai berikut:

1. Posisi awal *quadcopter* terletak pada titik XY(0,0).
2. *Quadrotor* bergerak menuju titik XY(2,2), dimana bergerak dengan mengubah sudut *pitch* dan *roll*.
3. *Quadrotor* bergerak menuju titik XY(2,-2), bergerak dengan mengubah sudut *roll* dan keaadaan sudut *pitch* tetap.

4. *Quadrotor* bergerak menuju titik XY(-2,-2), bergerak dengan mengubah sudut *pitch* dan keaadaan sudut *roll* tetap.
5. *Quadrotor* bergerak menuju titik XY(-2,2), bergerak dengan mengubah sudut *roll* dan keaadaan sudut *pitch* tetap.
6. *Quadrotor* bergerak menuju titik XY(2,2), bergerak dengan mengubah sudut *pitch* dan keaadaan sudut *roll* tetap.

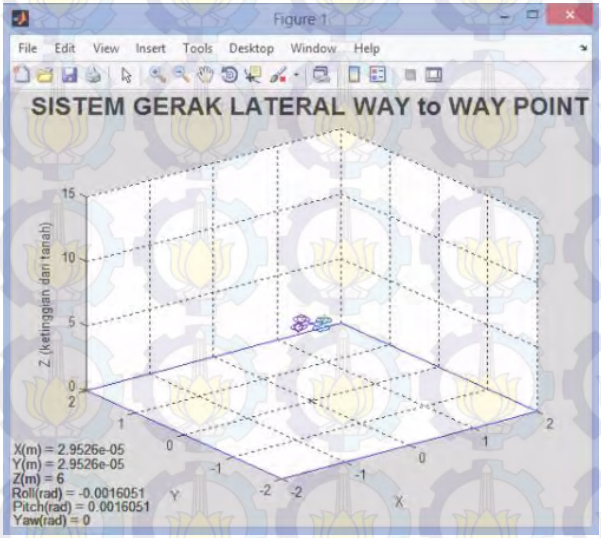
Hasil simulasi pada XY *Plot* berupa penampakan dua dimensi menunjukkan gerak translasi dari *quadcopter* untuk menuju titik koordinat yang telah ditentukan. Simulasi dua dimensi pada sistem gerak *way-to-way point* dapat menunjukkan performansi secara intuitif dari *quadcopter* seperti yang ditunjukkan pada Gambar 4.10. Dari hasil tersebut dapat terlihat bahwa ada kesalahan yang relatif kecil dari *quadcopter* untuk menuju titik yang telah ditentukan.



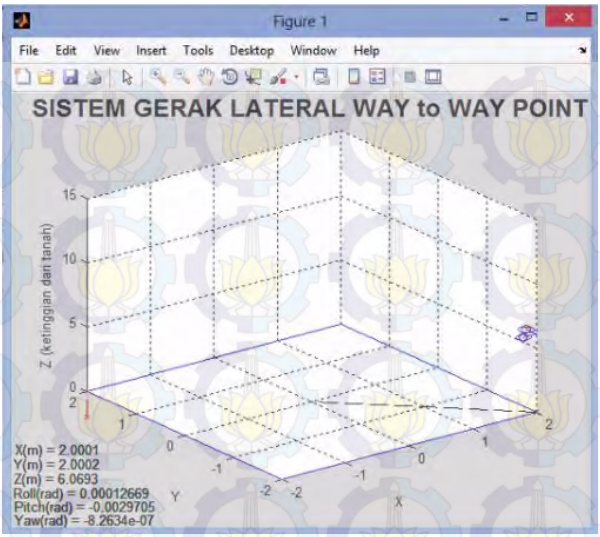
Gambar 4.10. Simulasi Sistem Gerak *Way-to-Way Point* pada XY Plot

Hasil simulasi tiga dimensi dapat menunjukkan pergerakan rotasi dan pergerakan translasi dari *quadcopter* secara visual untuk mencapai titik koordinat yang telah ditentukan. Simulasi tiga dimensi dapat menunjukkan performansi secara intuitif dari *quadcopter*. Selain itu, dalam simulasi tiga dimensi juga dapat ditampilkan hasil rancangan desain kontroler yang telah dibuat melalui nilai bilangan bulat yang ditampilkan saat *quadcopter* melakukan gerak lateral *tracking waypoint*. Nilai yang ditampilkan adalah sudut roll, sudut pitch, sudut yaw, sumbu X, sumbu Y, dan sumbu Z sesuai dari hasil persamaan model maematika *quadcopter* yang telah dijelaskan pada bab sebelumnya. Selain hasil dari respons sinyal kontrol, nilai yang ditampilkan pada hasil simulasi tiga

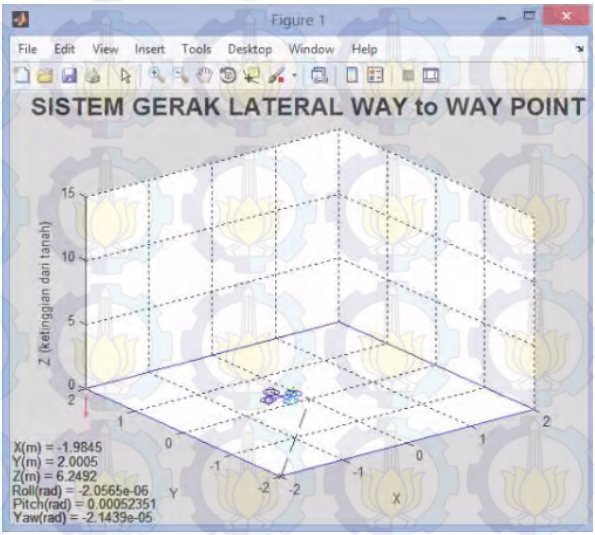
dimensi *quadcopter* juga dapat digunakan sebagai acuan dalam menentukan hasil rancangan kontroler yang telah dibuat. Hasil simulasi tiga dimensi *tracking waypoint quadcopter* ditampilkan pada Gambar 4.11 sampai Gambar 4.16.



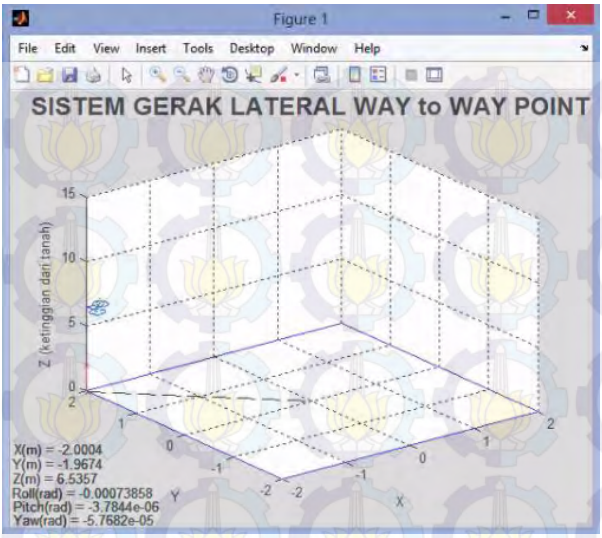
Gambar 4.11. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (0,0)



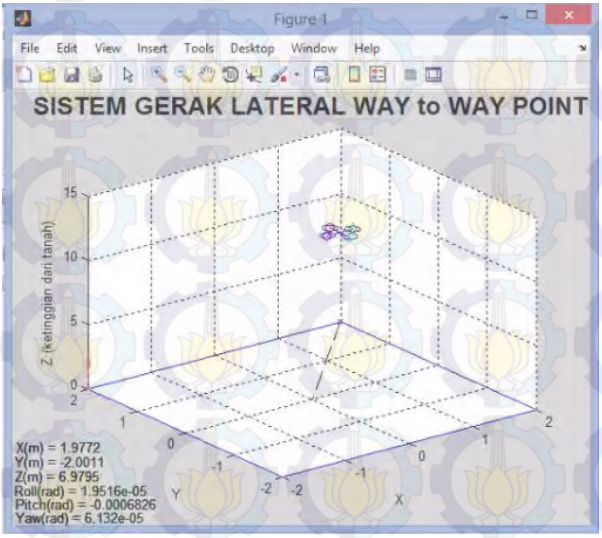
Gambar 4.12. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (2,2)



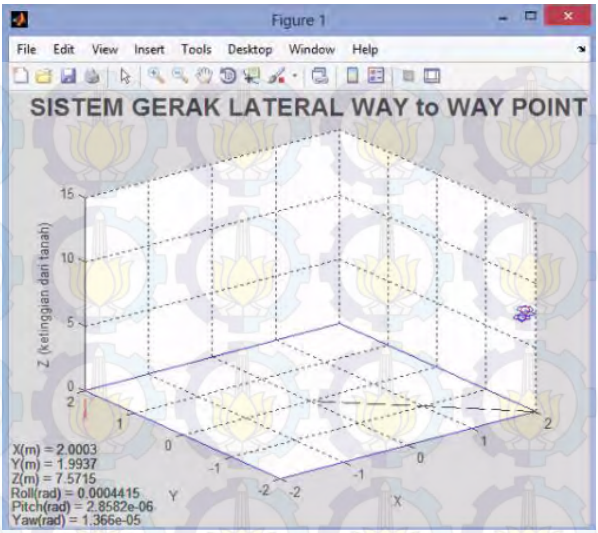
Gambar 4.13. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (-2,2)



Gambar 4.14. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (-2,-2)



Gambar 4.15. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (2,-2)

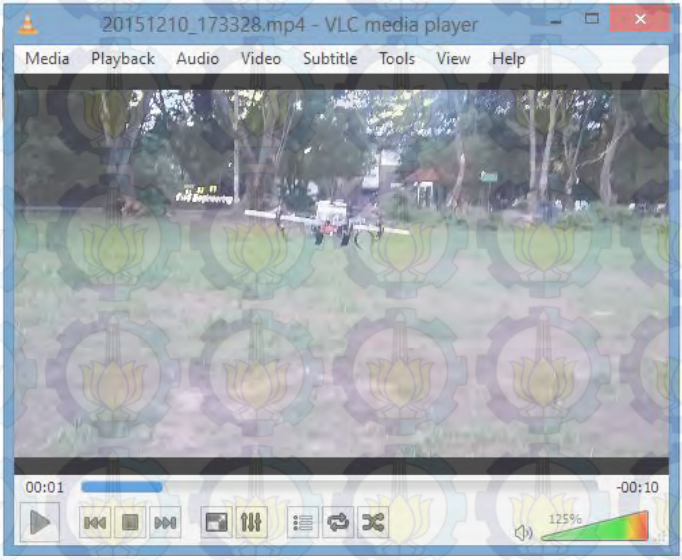


Gambar 4.16. Simulasi 3 Dimensi Sumbu X dan Y Pada Titik Koordinat (2,2)

4.6 Hasil Perencanaan Implementasi

Perencanaan implementasi masih belum bisa dilakukan karena keterbatasan dari kemampuan *flight controller* untuk dilakukan penelitian. Sampai saat ini uji coba terbang *quadcopter* dilakukan secara manual menggunakan *remote control* untuk melihat hasil rancangan desain mekanik yang telah dibuat. Hasil rancangan mekanik *quadcopter* seluruhnya ditampilkan pada gambar 4.17 dan hasil uji coba terbang ditampilkan pada Gambar 4.18.

Gambar 4.17 Hasil Rancang Bangun *Quadcopter*



Gambar 4.18 Hasil Uji Coba *Quadcopter*

Halaman ini sengaja dikosongkan

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan pada penelitian Tugas Akhir, diperoleh kesimpulan sebagai berikut :

1. Simulasi sistem gerak lateral *way-to-way point* menggunakan kontroler PID-*Fuzzy* mampu menghasilkan respons yang baik dengan atau tanpa *noise* (sinyal gangguan). *Quadcopter* mampu menuju titik koordinat yang ditentukan.
2. Respon hasil simulasi gerak translasi pada sumbu X dan Y masih terdapat kesalahan sebesar ± 0,01 cm dan ± 0,035 cm saat *quadcopter* berjalan menyusuri lintasan. Sedangkan respons waktu saat mencapai *steady-state* masih ada keterlambatan selama 2,2 detik.
3. Metode *tuning fuzzy* berhasil melakukan penalaan nilai parameter kontrol Kp dan Kd saat *quadcopter* melakukan gerak *tracking waypoint* yang di tampilkan pada Tabel 3.5 dan Tabel 3.6.
4. Implementasi uji coba untuk sistem gerak lateral *way-to-way point* belum bisa dilakukan karena keterbatasan *flight controller* APM 2.6.

5.2 Saran

Dari hasil penelitian yang dilakukan, untuk pengembangan berikutnya, disarankan beberapa hal berikut ini:

1. Pemodelan dan pemahaman mengenai mekanik, elektrik, dan ilmu pemrograman tentang *quadrotor* sangat dibutuhkan untuk membantu perancangan sistem yang lebih baik dan mendapatkan model matematika yang lebih akurat.
2. Perancangan mekanik yang lebih baik akan memudahkan implementasi algoritma kontrol yang akan digunakan dan dianalisis.
3. Penggunaan kotroler PID-*Fuzzy* nantinya dikembangkan lagi untuk keberlanjutan penelitian ini.

.

halaman ini sengaja dikosongkan

LAMPIRAN

Lampiran A : Program *Quadrotor_Plot*

```
% Copyright (C) 1993-2011, by Peter I. Corke
%
% This file is part of The Robotics Toolbox for Matlab (RTB).
%
% RTB is free software: you can redistribute it and/or modify
% it under the terms of the GNU Lesser General Public License as
published by
% the Free Software Foundation, either version 3 of the License, or
% (at your option) any later version.
%
% RTB is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied
warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the
% GNU Lesser General Public License for more details.
%
% You should have received a copy of the GNU Leser General Public
License
% along with RTB.  If not, see <http://www.gnu.org/licenses/>.
function [sys,x0,str,ts] = quadrotor_plot1(t,x,u,flag,s,plot,enable)
% Flyer plot, lovingly coded by Paul Pounds, first coded 17/4/02
% version 2 2004 added scaling and ground display
% version 3 2010 improved rotor rendering and fixed mirroring bug
%
% Displays X-4 flyer position and attitude in a 3D plot.
% GREEN ROTOR POINTS NORTH
% BLUE ROTOR POINTS EAST

% PARAMETERS
% s defines the plot size in meters
% swi controls flyer attitude plot; 1 = on, otherwise off.

% INPUTS
% 1 Center X position
% 2 Center Y position
% 3 Center Z position
```

```
% 4 Yaw angle in rad
% 5 Pitch angle in rad
% 6 Roll angle in rad

% OUTPUTS
%   None
ts = [-1 0];

switch flag,
  case 0
    [sys,x0,str,ts] = mdlInitializeSizes(ts,plot,enable); % Initialization
  case 3
    sys = mdlOutputs(t,u,s,plot,enable); % Calculate outputs
  case {1,2, 4, 9} % Unused flags
    sys = [];
  otherwise
    error(['unhandled flag = ',num2str(flag)]); % Error handling
end


% Initialize
function [sys,x0,str,ts] = mdlInitializeSizes(ts,plot,enable)
% Call simsizes for a sizes structure, fill it in, and convert it
% to a sizes array.
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 0;
sizes.NumInputs      = 6;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];          % Set str to an empty matrix.
ts = [0.05 0];

if enable == 1
    figure(plot);
    clf;
    %colordef(1,'none');
end
```

```
% End of mdlInitializeSizes.


function sys = mdlOutputs(t,u,s,plot,enable)
global a1s b1s

   name = strcat('flyer_movie',num2str((t/0.125)*2+1),'r.bmp'); %odds
   %name = strcat('flyer_movie',num2str(160-(t/0.125)*2),'l.bmp');
%evens


if size(a1s) == [0 0];
   a1s = [0 0 0 0];
   b1s = [0 0 0 0];
end

d = 0.2; %Hub displacement from COG
r = 0.1; %Rotor radius
N = 1;
S = 2;
E = 3;
W = 4;
D(:,N) = [d;0;0]; %displacements
D(:,S) = [-d;0;0];
D(:,E) = [0;d;0];
D(:,W) = [0;-d;0];
C(:,N) = [s(1)/4;0;0]; %Attitude center displacements
C(:,S) = [-s(1)/4;0;0];
C(:,E) = [0;s(1)/4;0];
C(:,W) = [0;-s(1)/4;0];

if enable == 1
   %draw ground
   figure(plot);
   clf;
   if length(s) == 1
      axis([-s s -s s 0 s]);
   else
      axis([-s(1) s(1) -s(1) s(1) 0 s(2)])
      s = s(1);
   end
```

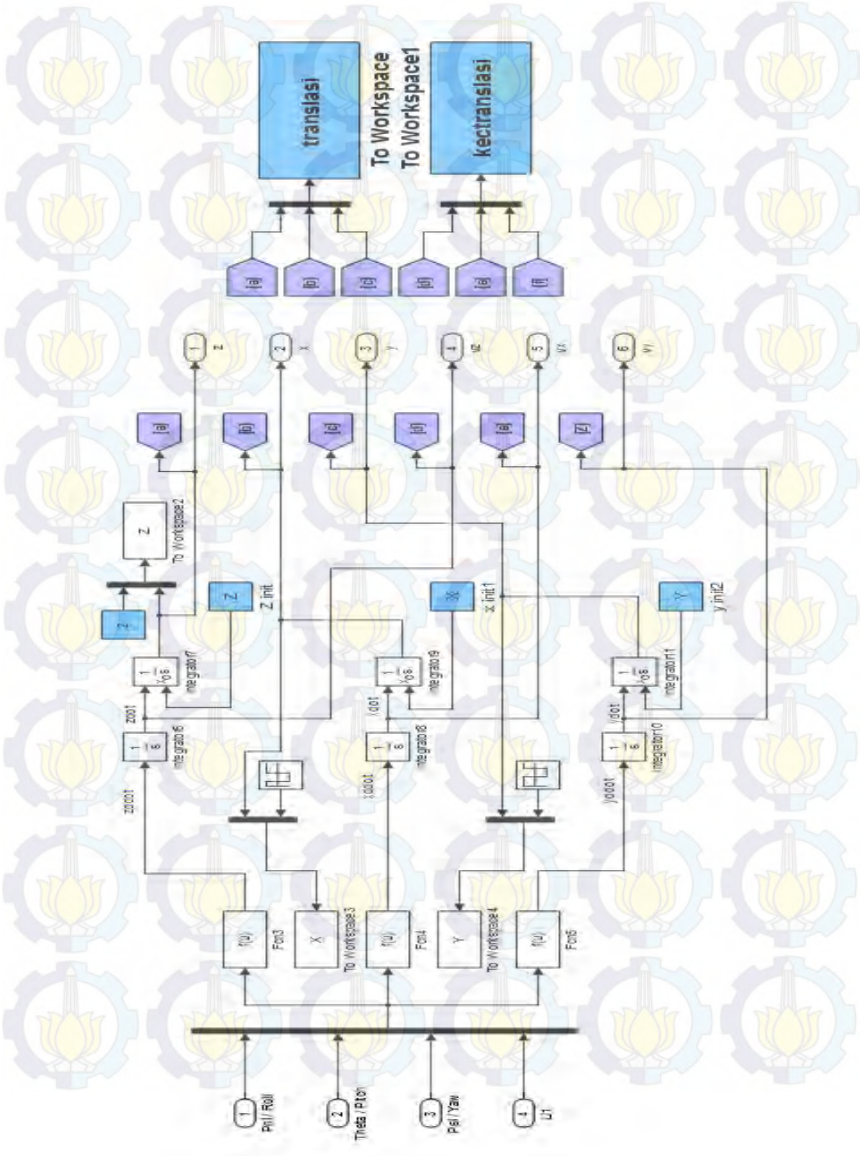
```
hold on;

plot3([-s -s],[s -s],[0 0],'-b')
plot3([-s s],[s s],[0 0],'-b') %garisbiruWEST
plot3([s -s],[-s -s],[0 0],'-b')
plot3([s s],[s -s],[0 0],'-b')%garisbiruNORTH
%plot3([s -s],[-s s],[0 0],'-b') %lintanglurus
%plot3([-s s],[-s s],[0 0],'-b') %lintangsilang

%READ STATE
z = [u(1);u(2);-u(3)];
n = [u(4);u(5);u(6)];

%PREPROCESS ROTATION MATRIX
phi = n(1);    %Euler angles
the = n(2);
psi = n(3);

R = [cos(the)*cos(phi) sin(psi)*sin(the)*cos(phi)-cos(psi)*sin(phi)
cos(psi)*sin(the)*cos(phi)+sin(psi)*sin(phi);   %BBF > Inertial rotation
matrix
     cos(the)*sin(phi) sin(psi)*sin(the)*sin(phi)+cos(psi)*cos(phi)
cos(psi)*sin(the)*sin(phi)-sin(psi)*cos(phi);
    -sin(the)       sin(psi)*cos(the)                  cos(psi)*cos(the)];

    %Manual Construction
    %Q3 = [cos(psi) -sin(psi) 0;sin(psi) cos(psi) 0;0 0 1];   %Rotation
mappings
    %Q2 = [cos(the) 0 sin(the);0 1 0;-sin(the) 0 cos(the)];
    %Q1 = [1 0 0;0 cos(phi) -sin(phi);0 sin(phi) cos(phi)];
    %R = Q3*Q2*Q1;    %Rotation matrix

%CALCULATE FLYER TIP POSITONS USING COORDINATE
FRAME ROTATION
F = [1 0 0;0 -1 0;0 0 -1];

%Draw flyer rotors
t = [0:pi/8:2*pi];
for j = 1:length(t)
   circle(:,j) = [r*sin(t(j));r*cos(t(j));0];
end
```

```
   title('SISTEM GERAK LATERAL WAY to WAY
POINT','FontSize',18,'FontName','arial','FontWeight','bold');
   for i = [N S E W]
      hub(:,i) = F*(z + R*D(:,i)); %points in the inertial frame

      q = 1; %Flapping angle scaling for output display - makes it easier
to see what flapping is occurring
      Rr = [cos(q*a1s(i))  sin(q*b1s(i))*sin(q*a1s(i))
cos(q*b1s(i))*sin(q*a1s(i));   %Rotor > Plot frame
            0            cos(q*b1s(i))            -sin(q*b1s(i));
           -sin(q*a1s(i)) sin(q*b1s(i))*cos(q*a1s(i))
cos(q*b1s(i))*cos(q*a1s(i))];

      tippath(:,:,i) = F*R*Rr*circle;

plot3([hub(1,i)+tippath(1,:,i)],[hub(2,i)+tippath(2,:,i)],[hub(3,i)+tippath(
3,:,i)],'b-')
      tinggi=u(3);
      roll_plot= u(6);
      pitch_plot= u(5);
      maju=u(1);
      minggir=u(2);
      yaw_plot=u(4);
      if tinggi<0.05
         tinggi=0;
      end
      if roll_plot<0.0001
         roll_plot= u(6);
      end
      if pitch_plot<0.0001
         pitch_plot= u(5);
      end

      textminggir=text(-4,0.85,['X(m) = ', num2str(maju)]); %posisi
tulisan X(m) pada grafik saat running
      textmaju=text(-4.33,0.45,['Y(m) = ', num2str(minggir)]); %posisi
tulisan Y(m) pada grafik saat running
      textheight=text(-4.63,0.05,['Z(m) = ', num2str(tinggi)]); %posisi
tulisan Z(m) pada grafik saat running
      textroll=text(-4.95,-0.35,['Roll(rad) = ', num2str(roll_plot)]);
%posisi tulisan roll(rad) pada grafik saat running
```

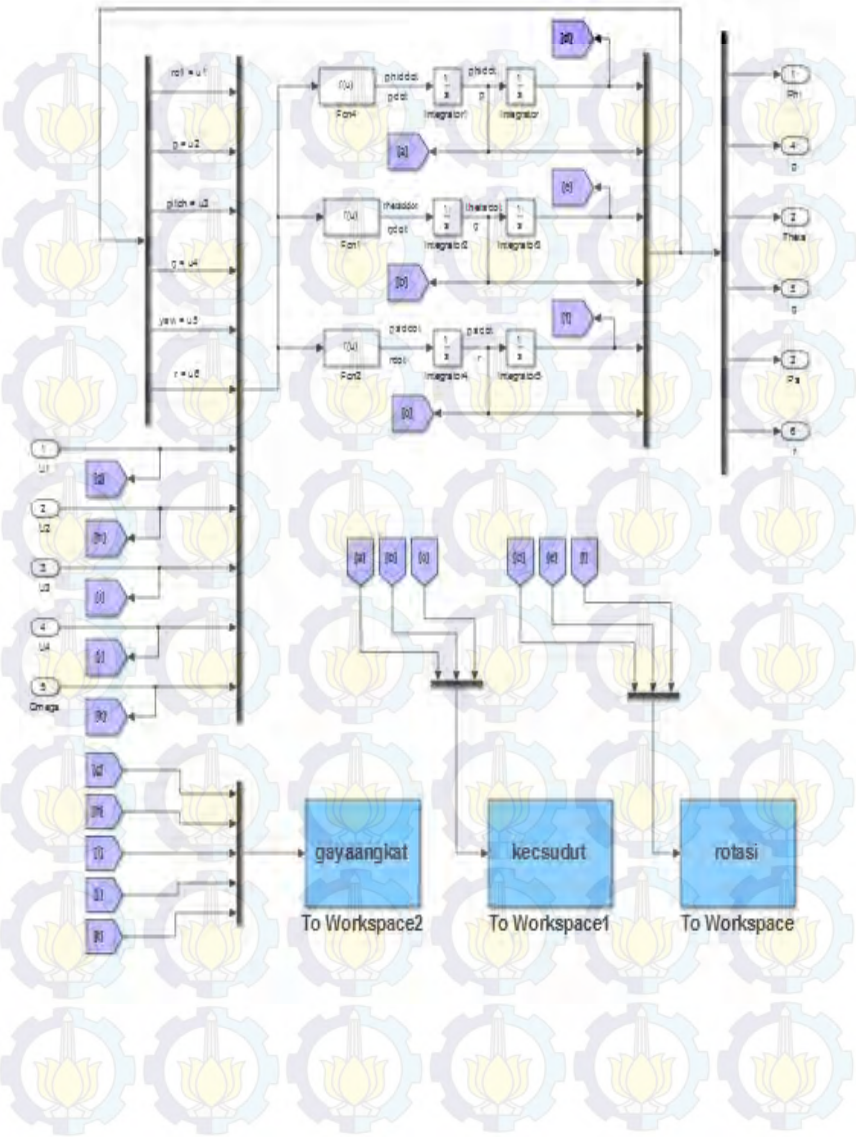
```
        textpitch=text(-5.25,-0.75,['Pitch(rad) = ', num2str(pitch_plot)]);
%posisi tulisan pitch(rad) pada grafik saat running
        textpitch=text(-5.55,-1.15,['Yaw(rad) = ', num2str(yaw_plot)]);
%posisi tulisan yaw(rad) pada grafik saat running
    end
    %Draw flyer
    plot3([hub(1,N) hub(1,S)],[hub(2,N) hub(2,S)],[hub(3,N) hub(3,S)],'-
d')
    plot3([hub(1,E) hub(1,W)],[hub(2,E) hub(2,W)],[hub(3,E)
hub(3,W)],'-b')
    plot3([hub(1,N)],[hub(2,N)],[hub(3,N)],'og')
    plot3([hub(1,S)],[hub(2,S)],[hub(3,S)],'or')
    plot3([hub(1,E)],[hub(2,E)],[hub(3,E)],'oc')   %isi lingkaran dalam
    plot3([hub(1,W)],[hub(2,W)],[hub(3,W)],'or')

    %Tracking lines
    plot3([z(1) 0],[-z(2) 0],[0 0],'--k')   %titik2trajektoriX
    plot3([z(1)],[-z(2)],[0],'xk')      %titik2trajektoriY
    plot3([-s -s],[s s],[-z(2) 0],'--r') %titik2trajektoriZ
    plot3([-s],[s],[-z(2)],'xr')
    xlabel('X');
    ylabel('Y');
    zlabel('Z (ketinggian dari tanah)');
    grid on
    %mov=avifile('quad.avi','Compression','None','Quality',100,'fps',10)
    %video=figure(plot);
end


%saveas(1,name);

sys = [];

% End of mdlOutputs.
```

Lampiran B : Diagram Blok Simulasi Pada MATLAB

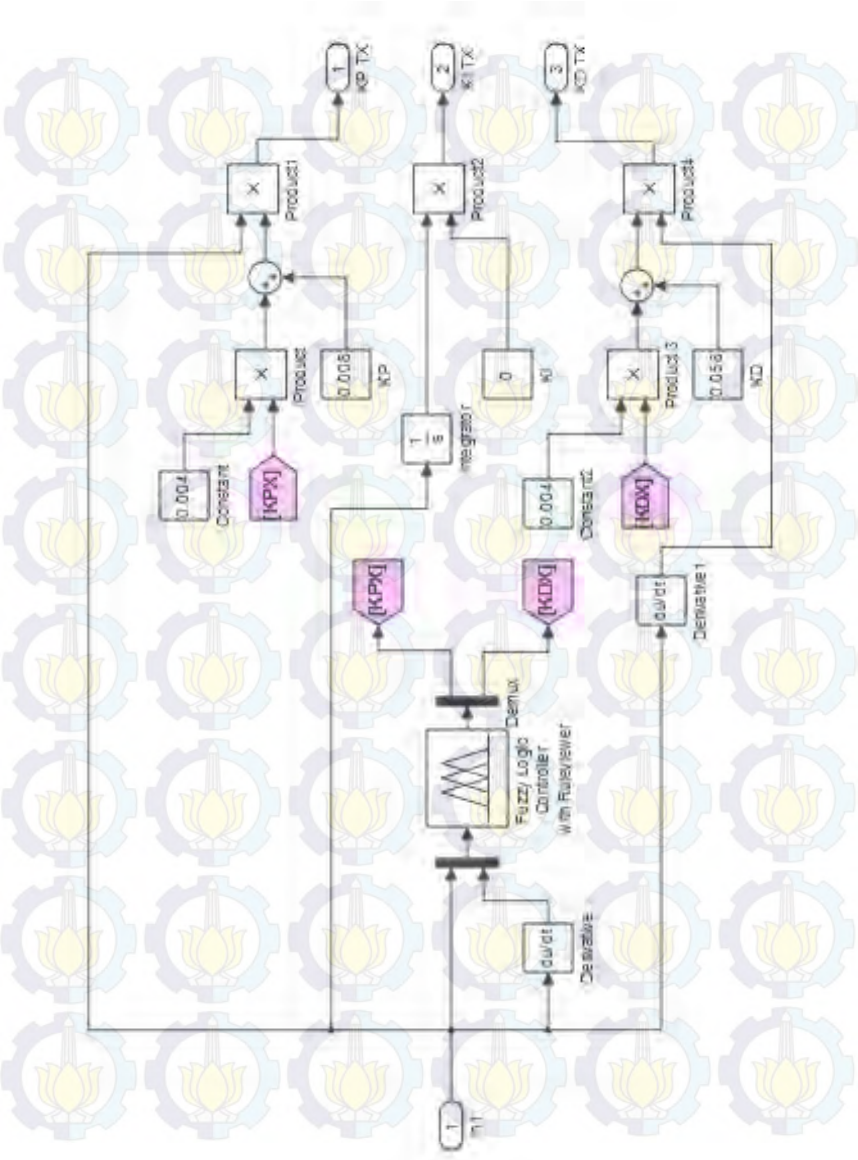
B.1 Persamaan Translasi



B.2 Persamaan Rotasi

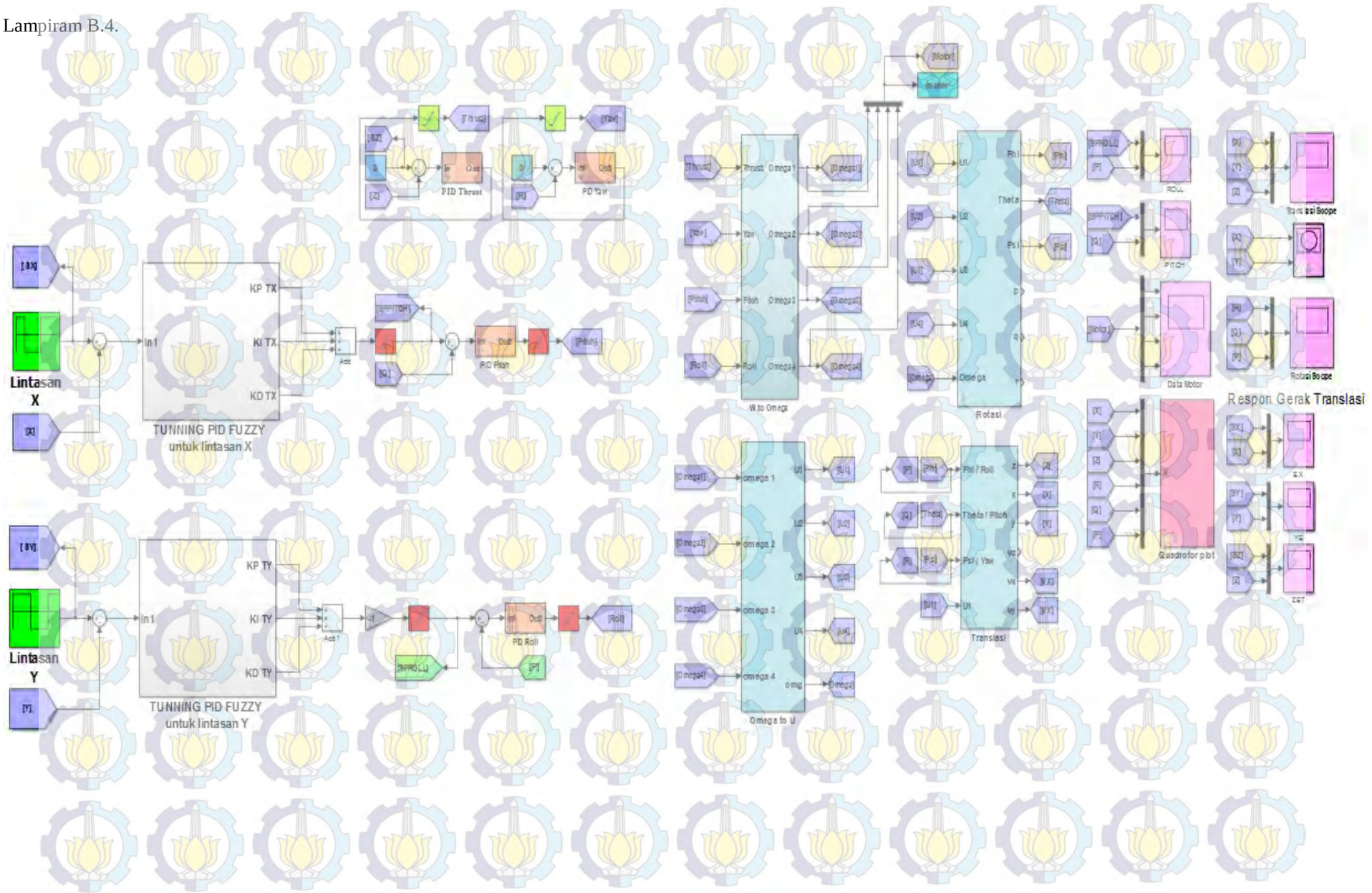


B.3 *Auto Tuning Fuzzy* untuk Penalaan Parameter PID

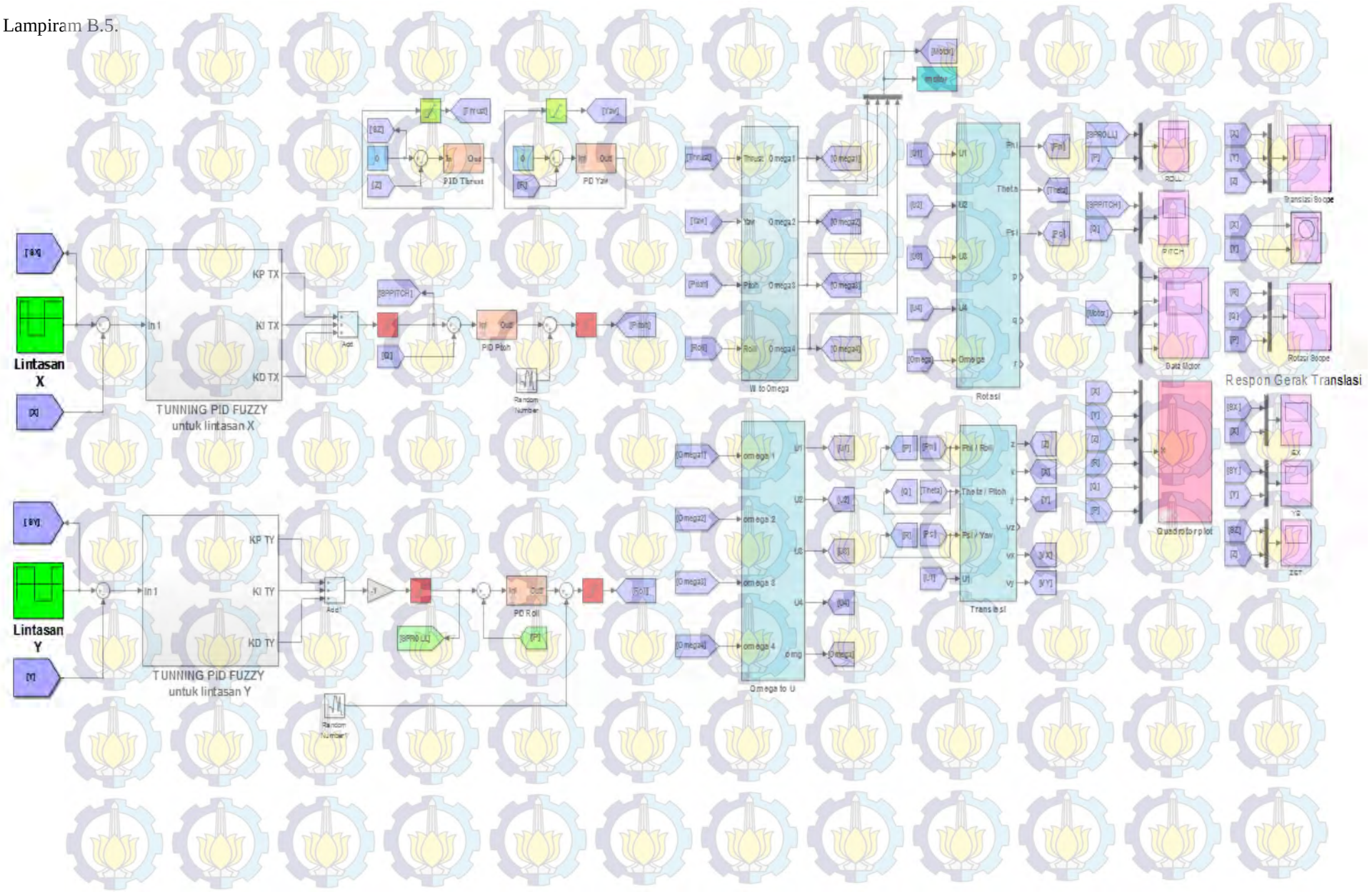


halaman ini sengaja dikosongkan

Lampiran B.4.


Lintasan X
Lintasan Y
KP TX
KI TX
KD TX
KP TY
KI TY
KD TY
TUNNING PID FUZZY untuk lintasan X
TUNNING PID FUZZY untuk lintasan Y
PID Thrust
PID Yaw
PID Pitch
PID Roll
Wto Omega
Omega to U
Rotasi
Translasi
ROLL
PITCH
Data Motor
Quadrofor plot
Respon Gerak Translasi

Lampiram B.5.


Lintasan X
Lintasan Y
KP TX
KI TX
KD TX
KP TY
KI TY
KD TY
TUNNING PID FUZZY
untuk lintasan X
TUNNING PID FUZZY
untuk lintasan Y
PID Thrust
PID Yaw
PID Pitch
PID Roll
W to Omega
Omega to U
Rotasi
Translasi
Respon Gerak Translasi

DAFTAR PUSTAKA

[1]. Luukkonen, Teppo, **"Modelling and Control of Quadcopter"**, Aalto University, Espoo, 2011.

[2]. Gamayanti, Nurlita, "**Diktat Mata Kuliah Dasar Sistem Pengaturan Pengaturan**", Jurusan Teknik Elektro FTI-ITS, Surabaya, 2010.

[3]. Cheng-Hao Huang, "**Fuzzy Control Applications to Robotic Systems**". Jhongli, Taiwan: National Central University, 2011.

[4]. _____."**Turnigy Talooon Frame for Quadcopter**", www.hobbymania.co.za. Diakses pada tanggal 06 Desember 2015

[5]. _____."**Turnigy Propeller for Quadcopter 10 x 4.5 cm**", www.readytoflyquads.com. Diakses pada tanggal 04 Desember 2015

[6]. _____."**Ardupilot Mega 2.6 Quadcopter Flight Controller**", www.readymaderc.com. Diakses pada tanggal 25 November 2015

[7]. _____,"**Turnigy 9X 9Ch Transmitter With Module 8ch Receiver Mode 2 v2 Firmware**" www.hobbyking.com. Diakses pada tanggal 06 Desember 2015

[8]. _____."**Electronic Speed Controller Bulletproof 30 A quadcopter**", www.quadcopters.co.uk. Diakses pada tanggal 28 November 2015

[9]. _____."**Sunnysky V2216 Brushless Motor 900 kV**", www.multirotorsuperstore.com. Diakses pada tanggal 24 November 2015

[10]. _____."**Ardupilot Flight Controller For Quadcopter**", www.hobbyking.com. Diakses pada tanggal 25 November 2015

[11]. _____."**LiPo Baterai 5800 Mah**",www.readymaderc.com. Diakses pada tanggal 28 November 2015

[12]. Tommaso Bresciani, "**Modelling, Identification and Control of a Quadrotor Helicopter**". Department of Automatic Control Lund University, October 2008.

[13]. Ogata, katsuhiko "**Teknik Kontrol Automatik – terjemahan Edi Laksono**" Erlangga, Jakarta 1991.

[14]. Bouktir, Y., Haddad, M., Chettibi, T., "Trajectory Planning for A Quadrotor Helicopter", **16th Mediterranean Conference on Control and Automation**, Ajaccio, France, Juni, 2008.

[15]. G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "*Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment*" **Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit**, Aug.2007.

[16]. H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," **IEEE International Conference on Robotics and Automation**, pp. 3277 - 3282, May 2009.

[17]. I. C. Dikmen, A. Arısoy, and H. Temelta¸s, "Attitude control of a quadrotor," **4th International Conference on Recent Advances in Space Technologies**, pp. 722-727, 2009.

[18]. Zulfatman dan M. F. Rahmat, "Application of Self-Tuning Fuzzy PID Kontroller on Industrial Hydraulic Actuator Using Sistem Identification Approach," **International Journal on Smart Sensing and Intelligent Systems**, Vol. 2, No.2, June 2009

RIWAYAT PENULIS

Rheco Ari Prayogo yang memiliki nama panggilan Rheco dilahirkan di Probolinggo, 28 Oktober 1991. Merupakan putra pertama dari pasangan Seno dan Ekawati. Penulis menempuh pendidikan SD, SMP, dan SMA di Sekolah Taruna Dra Zulaeha dan lulus pada tahun 2004, 2007, dan 2010. Setelah menyelesaikan pendidikan SMA, penulis melanjutkan studinya di Institut Teknologi Sepuluh Nopember Surabaya tepatnya pada jurusan Teknik Elektro Prodi Diploma Teknik Elektro Bidang Studi Komputer Kontrol dan lulus pada tahun 2013. Selanjutnya penulis meneruskan studi sarjana di Teknik Elektro ITS, dengan mengambil fokus pada bidang studi Teknik Sistem Pengaturan. Pada bulan Januari 2016 penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro dari Institut Teknologi Sepuluh Nopember Surabaya.

halaman ini sengaja dikosongkan